User Guide

AWS Clean Rooms

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Clean Rooms: User Guide

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is AWS Clean Rooms? ... 1
Are you a first-time AWS Clean Rooms user? ... 1
How AWS Clean Rooms works ... 2
Related services ... 2
AWS services ... 2
Third-party services ... 4
Accessing AWS Clean Rooms ... 4
Pricing for AWS Clean Rooms ... 5
Billing for AWS Clean Rooms ... 5
Analysis rules ... 5
Analysis rule types ... 6
Aggregation analysis rule ... 9
List analysis rule ... 28
Custom analysis rule ... 36
ID mapping table analysis rule ... 82
AWS Clean Rooms Differential Privacy ... 92
Differential privacy ... 93
How Differential Privacy in AWS Clean Rooms works ... 93
Differential privacy policy ... 94
SQL capabilities ... 96
SQL query tips and examples ... 126
Limitations ... 127
AWS Clean Rooms ML ... 128
AWS Clean Rooms ML terminology ... 129
How AWS Clean Rooms ML works with AWS models ... 130
How AWS Clean Rooms ML works with custom models ... 131
AWS models in Clean Rooms ML ... 133
Custom models in Clean Rooms ML ... 140
Cryptographic computing ... 149
Considerations ... 150
Supported file and data types ... 153
Column names ... 158
Column types ... 158
Parameters ... 161

Optional flags ... 166
Queries with C3R ... 168
Guidelines ... 169
Setting up AWS Clean Rooms ... 196
Sign up for an AWS account ... 196
Set up service roles for AWS Clean Rooms ... 196
Create an IAM role for a collaboration member ... 196
Create a service role to read data from Amazon S3 ... 197
Create a service role to read data from Amazon Athena ... 201
Create a service role to read data from Snowflake ... 205
Create a service role to read code from an S3 bucket (PySpark analysis template role) ... 209
Create a service role to write results of a PySpark job ... 212
Create a service role to receive results ... 215
Set up service roles for AWS Clean Rooms ML ... 219
Set up service roles for lookalike modeling ... 219
Set up service roles for custom modeling ... 233
Collaborations and memberships ... 249
Creating a collaboration ... 250
Creating a collaboration for queries ... 250
Creating a collaboration for queries and jobs ... 261
Creating a collaboration for ML modeling ... 271
Creating a membership and joining a collaboration ... 281
Updating a membership ... 287
Update analysis logging settings ... 287
Update default result settings ... 288
Update payment configuration ... 289
Editing collaborations ... 290
Edit collaboration name and description ... 291
Update collaboration analytics engine ... 291
Turn off log storage ... 292
Edit collaboration logs settings ... 292
Edit collaboration tags ... 293
Edit membership tags ... 294
Add a new member ... 295
Edit existing member abilities ... 295
Edit collaboration auto-approval settings ... 295

Edit associated table tags ... 295
Edit analysis template tags ... 296
Edit differential privacy policy tags ... 296
Change requests ... 297
Add a new member to a collaboration ... 297
Update member abilities ... 298
Update ML member abilities ... 300
Update payer configurations ... 301
Edit collaboration auto-approval settings ... 302
Deleting collaborations ... 303
Viewing collaborations ... 303
Inviting members ... 304
Monitoring members ... 304
Adding members ... 305
Removing members ... 306
Leaving a collaboration ... 307
Data tables ... 308
Data formats ... 309
Supported data formats for PySpark jobs ... 309
Supported data formats for SQL queries ... 309
Supported data types ... 310
File compression types for AWS Clean Rooms ... 311
Server-side encryption for AWS Clean Rooms ... 311
Apache Iceberg tables ... 312
Supported data types for Iceberg tables ... 313
Preparing data tables ... 313
Preparing data tables in Amazon S3 ... 314
Preparing data tables in Amazon Athena ... 317
Preparing data tables in Snowflake ... 319
Preparing encrypted data tables ... 321
Step 1: Complete the prerequisites ... 322
Step 2: Download the C3R encryption client ... 323
Step 3: (Optional) View available commands in the C3R encryption client ... 323
Step 4: Generate an encryption schema for a tabular file ... 324
Step 5: Create a shared secret key ... 331
Step 6: Store the shared secret key in an environment variable ... 332

Step 7: Encrypt data ... 333
Step 8: Verify data encryption ... 334
(Optional) Create a schema (advanced users) ... 335
Decrypting data tables ... 344
Configured tables ... 346
Creating a configured table ... 347
Amazon S3 data source ... 347
Amazon Athena data source ... 350
Snowflake data source ... 353
Adding an analysis rule to a configured table ... 358
Adding an aggregation analysis rule to a table (guided flow) ... 358
Adding a list analysis rule to a table (guided flow) ... 362
Adding a custom analysis rule to a table (guided flow) ... 364
Adding analysis rule to a table (JSON editor) ... 368
Next steps ... 370
Associating a configured table to a collaboration ... 370
Data access budget ... 370
Associate a configured table ... 371
Configuring a data access budget ... 377
Adding a data access budget to an existing associated table ... 377
Viewing a data access budget ... 379
Editing a data access budget ... 379
Deleting a data access budget ... 381
Adding a collaboration analysis rule to a configured table ... 382
Configuring differential privacy policy (optional) ... 383
Viewing differential privacy usage logs ... 384
Editing a differential privacy policy ... 385
Deleting a differential privacy policy ... 386
Viewing the calculated differential privacy parameters ... 386
Viewing tables and analysis rules ... 387
Editing a configured table ... 388
Editing configured table tags ... 389
Editing the configured table analysis rule ... 389
Deleting the configured table analysis rule ... 390
Configured table disallowed columns ... 390
Editing configured table associations ... 394

Disassociating configured tables ... 395
Deleting a configured table ... 395
AWS Entity Resolution in AWS Clean Rooms ... 397
ID namespaces ... 398
Creating and associating a new ID namespace ... 398
Associating an existing ID namespace ... 401
Editing ID namespace associations ... 403
Disassociating ID namespace associations ... 404
ID mapping tables ... 405
Creating and populating a new ID mapping table ... 406
Populating an existing ID mapping table ... 413
Editing an ID mapping table ... 414
Deleting an ID mapping table ... 415
Analysis templates ... 417
SQL analysis templates ... 417
Creating a SQL analysis template ... 418
Reviewing a SQL analysis template ... 421
PySpark analysis templates ... 422
Security ... 423
Limitations ... 423
Best practices ... 424
Creating a user script ... 425
Working with parameters in PySpark analysis templates ... 429
Creating a virtual environment (optional) ... 433
Storing a user script and virtual environment in S3 ... 434
Creating a PySpark analysis template ... 436
Reviewing a PySpark analysis template ... 440
Troubleshooting PySpark analysis templates ... 443
Troubleshooting your code ... 443
Analysis template job doesn't start ... 444
Analysis template job starts but fails during processing ... 445
Virtual environment setup fails ... 447
Intermediate tables ... 449
Creating an intermediate table ... 452
Populating an intermediate table ... 454
Configuring an analysis rule for an intermediate table ... 455

Deleting an intermediate table ... 457
Other deletion triggers ... 457
Analysis ... 459
Running SQL queries ... 459
Prerequisites ... 460
Spark properties configuration for SQL queries ... 461
Querying configured tables ... 462
Querying ID mapping tables ... 484
Querying configured tables using an analysis template ... 487
Querying with the analysis builder ... 506
Viewing the impact of differential privacy ... 512
Viewing recent queries ... 513
Viewing query details ... 514
Exporting query analysis logs ... 514
Running PySpark jobs ... 527
Running a job using an analysis template ... 528
Viewing recent jobs ... 549
Viewing job details ... 549
Using an intermediate table in analyses ... 550
Analysis results ... 552
Receiving query results ... 553
Receiving job results ... 554
Editing default values for query results settings ... 555
Editing default values for job results settings ... 557
Using query output in other AWS services ... 557
Custom modeling ... 559
Privacy-enhanced synthetic dataset generation ... 560
Considerations for synthetic data generation ... 561
Creating and joining the collaboration ... 564
Creating a collaboration for machine learning ... 564
Joining a collaboration ... 568
Contributing training data ... 570
Configuring a model algorithm ... 575
Associating the configured model algorithm ... 578
Creating an ML input channel ... 583
Creating a trained model ... 604

Using incremental training ... 608
Using distributed training ... 612
Exporting model artifacts ... 616
Running inference on a trained model ... 618
ML modeling for training data providers ... 621
Importing training data ... 622
Creating a lookalike model ... 623
Configuring a lookalike model ... 624
Associating a configured lookalike model ... 625
Updating a configured lookalike model ... 626
ML modeling for seed data providers ... 628
Creating a lookalike segment ... 628
Exporting a lookalike segment ... 646
Troubleshooting ... 648
Using AWS MCP to troubleshoot Clean Rooms issues ... 648
Common issues ... 648
One or more tables referenced by the query is not accessible by its associated service role. The table/role owner must grant the service role access to the table. ... 649
One of the underlying datasets has an unsupported file format. ... 649
Query results are not as expected when using Cryptographic Computing for Clean Rooms. ... 649
AWS Clean Rooms Spark SQL: Missing partition data ... 650
Security ... 651
Data protection ... 652
Encryption at rest ... 653
Encryption in transit ... 653
Encrypting underlying data ... 654
Key policy ... 654
Encryption for intermediate tables ... 660
Using service-linked roles ... 663
Service-linked role permissions for AWS Clean Rooms ... 664
Creating a service-linked role for AWS Clean Rooms ... 664
Editing a service-linked role for AWS Clean Rooms ... 665
Deleting a service-linked role for AWS Clean Rooms ... 665
Supported Regions for AWS Clean Rooms service-linked roles ... 665
Data retention ... 665

Best practices ... 666
Best practices with AWS Clean Rooms ... 667
Best practices for using analysis rules in AWS Clean Rooms ... 667
Identity and Access Management ... 668
Audience ... 669
Authenticating with identities ... 669
Managing access using policies ... 671
How AWS Clean Rooms works with IAM ... 673
Identity-based policy examples ... 678
AWS managed policies ... 681
Troubleshooting ... 688
Cross-service confused deputy prevention ... 690
IAM behaviors for AWS Clean Rooms ML ... 692
IAM behaviors for Clean Rooms ML Custom Models ... 696
Compliance validation ... 698
Resilience ... 698
Infrastructure security ... 699
Network security ... 699
AWS PrivateLink ... 700
Considerations ... 700
Create an interface endpoint ... 700
Monitoring ... 702
Analysis logging in AWS Clean Rooms ... 703
Example analysis logs ... 707
Delivery, recipients, and log groups ... 709
Recommended actions for query and job logs ... 710
Detailed monitoring with CloudWatch in AWS Clean Rooms ... 710
Types of metrics ... 711
Metric dimensions ... 712
Who can access metrics ... 713
Pricing ... 713
CloudTrail logs ... 713
AWS Clean Rooms information in CloudTrail ... 714
Understanding AWS Clean Rooms log file entries ... 714
Example AWS Clean Rooms CloudTrail events ... 715
Integrating AWS Clean Rooms into EDAs ... 718

AWS Clean Rooms events ... 719
Routing AWS Clean Rooms events ... 724
Events detail reference ... 725
Cost Allocation Tagging ... 734
Taggable Resources for Cost Allocation ... 734
AWS CloudFormation resources ... 736
AWS Clean Rooms and CloudFormation templates ... 736
Learn more about CloudFormation ... 736
Quotas ... 738
AWS Clean Rooms quotas ... 738
AWS Clean Rooms resource parameter limits ... 754
AWS Clean Rooms ML quotas ... 754
Clean Rooms ML API throttling quotas ... 782
Document history ... 789
Glossary ... 807
Aggregation analysis rule ... 807
Analysis rules ... 807
Analysis template ... 807
Base table ... 807
C3R encryption client ... 808
Cleartext column ... 808
Collaboration ... 808
Collaboration creator ... 808
Configured table ... 809
Custom analysis rule ... 809
Decryption ... 809
Differential privacy ... 809
Encryption ... 810
Fingerprint column ... 810
ID mapping workflow method ... 810
ID mapping table ... 810
ID mapping table analysis rule ... 810
ID mapping workflow ... 811
ID namespace ... 811
ID namespace association ... 811
Intermediate table ... 811

Job 811
List analysis rule 811
Lookalike model 812
Lookalike segment 812
Member 812
Member who can query 812
Member who can run queries and jobs 812
Member who can receive results 813
Member paying for query compute costs 813
Member paying for query and job compute costs 813
Membership 814
Sealed column 814
Seed data 814
Spark analytics engine 814
Query 814

What is AWS Clean Rooms?

AWS Clean Rooms helps you and your partners analyze and collaborate on your collective datasets to gain new insights without revealing underlying data to one another. AWS Clean Rooms is a secure collaboration workspace, where you create your own clean rooms in minutes, and analyze your collective datasets with just a few steps. You choose the partners with whom you want to collaborate, select their datasets, and configure privacy-enhancing controls for those partners.

With AWS Clean Rooms, you can collaborate with thousands of companies already using AWS. Collaboration doesn't require moving data out of AWS or loading it into another cloud services provider. When you run queries or jobs, AWS Clean Rooms reads data from that data's original location and applies built-in analysis rules to help you maintain control over that data.

AWS Clean Rooms provides built-in data access controls and audit support controls that you can configure. These controls include:

- Analysis rules to restrict SQL queries and provide output constraints.
- Cryptographic Computing for Clean Rooms to keep data encrypted, even as queries are processed, to comply with stringent data handling policies.
- Analysis logs to review queries and jobs in AWS Clean Rooms and help support audits.
- Differential privacy to protect against user-identification attempts. AWS Clean Rooms Differential Privacy is a fully-managed capability that protects the privacy of your users with mathematically-backed techniques and intuitive controls that you can apply in a few steps.
- AWS Clean Rooms ML to allow two parties to identify similar users in their data without the need to share their data with each other. The first party creates and configures a lookalike model from their training data. Then, seed data is brought to the collaboration to create a lookalike segment that resembles the training data.

The following video explains more about AWS Clean Rooms.

AWS Clean Rooms

Are you a first-time AWS Clean Rooms user?

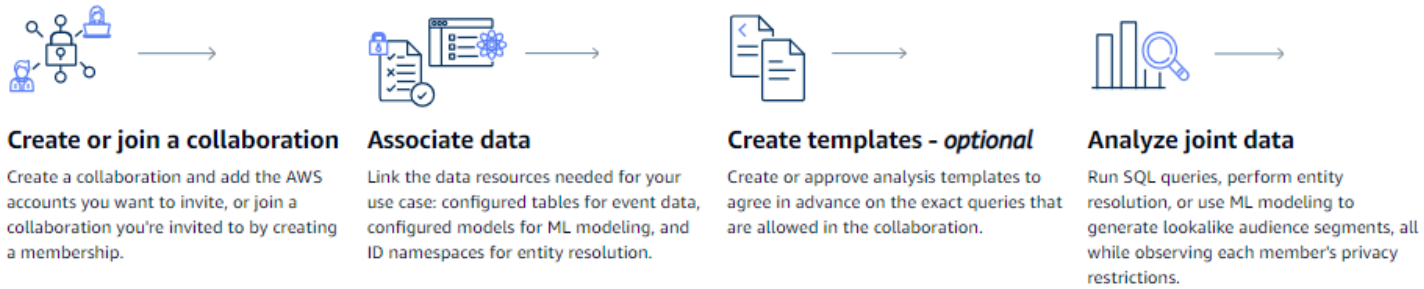
If you are a first-time user of AWS Clean Rooms, we recommend that you begin by reading the following sections:

- How AWS Clean Rooms works
- Accessing AWS Clean Rooms
- Setting up AWS Clean Rooms
- AWS Clean Rooms Glossary

How AWS Clean Rooms works

In AWS Clean Rooms, you create a collaboration and add the AWS accounts that you want to invite, or create a membership to join a collaboration that you've been invited to. You then link the data resources needed for your use case: configured tables for event data, configured models for ML modeling, or ID namespaces for entity resolution. You have the option to create or approve analysis templates to agree in advance on the exact queries and jobs that you want to allow in a collaboration. Finally, you analyze the joint data by running SQL queries or PySpark jobs on the configured tables, performing entity resolution in ID mapping tables, or using ML modeling to generate lookalike audience segments.

The following diagram shows how AWS Clean Rooms works.



Related services

AWS services

The following AWS services are related to AWS Clean Rooms:

- **Amazon Athena**

  Collaboration members can store data that they bring into AWS Clean Rooms as AWS Glue Data Catalog views in Amazon Athena. For more information, see the following topics:

  For more information, see the following topics:

Preparing data tables for queries in AWS Clean Rooms

Creating a configured table – Amazon Athena data source

What is Amazon Athena? in the *Amazon Athena User Guide*

- **CloudFormation**

  Create the following resources in CloudFormation: collaborations, configured tables, configured table associations, and memberships

  For more information, see Creating AWS Clean Rooms resources with AWS CloudFormation.

- **AWS CloudTrail**

  Use AWS Clean Rooms with CloudTrail logs to enhance your analysis of AWS service activity.

  For more information, see Logging AWS Clean Rooms API calls using AWS CloudTrail.

- **AWS Entity Resolution**

  Use AWS Clean Rooms with AWS Entity Resolution to perform entity resolution.

  For more information, see AWS Entity Resolution in AWS Clean Rooms.

- **AWS Glue**

  Collaboration members can create AWS Glue tables from their data in Amazon S3 for use in AWS Clean Rooms.

  For more information, see the following topics:

  Preparing data tables for queries in AWS Clean Rooms

  What is AWS Glue? in the *AWS Glue Developer Guide*

- **Amazon Simple Storage Service** (Amazon S3)

  Collaboration members can store data that they bring into AWS Clean Rooms in Amazon S3.

  For more information, see the following topics:

  Preparing data tables for queries in AWS Clean Rooms

  Creating a configured table – Amazon S3 data source

What is Amazon S3? in the *Amazon Simple Storage Service User Guide*

- **AWS Secrets Manager**

  Collaboration members can create secrets to access and read data stored in Snowflake.

  For more information, see the following topics:

  Create a service role to read data from Snowflake

  Preparing data tables for queries in AWS Clean Rooms

  What is AWS Secrets Manager? in the *AWS Secrets Manager User Guide*

Third-party services

The following third-party service is related to AWS Clean Rooms:

- **Snowflake**

  Collaboration members can store data that they bring into AWS Clean Rooms in a Snowflake warehouse.

  For more information, see the following topics:

  Preparing data tables for queries in AWS Clean Rooms

  Creating a configured table – Snowflake data source

Accessing AWS Clean Rooms

You can access AWS Clean Rooms through the following options:

- Directly through the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
- Programmatically through the AWS Clean Rooms API. For more information, see the *AWS Clean Rooms API Reference*.

Pricing for AWS Clean Rooms

For pricing information, see AWS Clean Rooms Pricing.

> **Note**
>
> For collaboration members who associated data that is stored in Snowflake, you will be charged by their respective data warehouse provider or cloud provider for both data egress and compute each time a query is run that uses data stored in those locations.

Billing for AWS Clean Rooms

AWS Clean Rooms gives the collaboration creator the ability to designate which member is paying for query or job compute costs in the collaboration.

In most cases, the member who can query and the member paying for query compute costs are the same. However, if the member who can query and the member paying for query compute costs are different, then, when the member who can query runs queries against their own membership resource, the membership resource of the member paying for query compute costs is billed.

The member paying for query compute costs doesn't see any event for queries being run in their CloudTrail Event history because the payer is neither the one running the queries nor the owner of the resource against which the queries are run. However, the payer does see charges generated on their membership resource for all queries run by the member who can run queries in the collaboration.

For more information about how to create a collaboration and configure the member paying for query compute costs, see Creating a collaboration.

Analysis rules in AWS Clean Rooms

As part of enabling a table to use in AWS Clean Rooms for collaboration analysis, the collaboration member must configure an *analysis rule*.

An analysis rule is a privacy-enhancing control that each data owner sets up on a configured table. An analysis rule determines how the configured table can be analyzed.

The analysis rule is an account-level control on the configured table (an account-level resource) and is enforced in any collaboration where the configured table is associated. If there is no analysis rule configured, the configured table can be associated to collaborations but it can't be queried. Queries can only reference configured tables with the same analysis rule type.

To configure an analysis rule, you first select a type of analysis and then specify the analysis rule. For both steps, you should consider the use case you want to enable and how you want to protect your underlying data.

AWS Clean Rooms enforces the more restrictive controls across all configured tables referenced in a query.

The following examples illustrate the restrictive controls.

Example Restrictive control: Output constraint

- Collaborator A has an output constraint on the identifier column of 100.
- Collaborator B has an output constraint on the identifier column of 150.

  An aggregation query that references both configured tables requires at least 150 distinct values of identifier within an output row for it to be displayed in the query output. The query output doesn't indicate that results are removed because of the output constraint.

Example Restrictive control: Analysis template not approved

- Collaborator A has allowed an analysis template with a query that references configured tables from Collaborator A and Collaborator B in their custom analysis rule.
- Collaborator B hasn't allowed the analysis template.

  Because Collaborator B hasn't allowed the analysis template, the member who can query can't run that analysis template.

Analysis rule types

There are three types of analysis rules: aggregation, list and custom. The following tables compare the analysis rule types. Each type has a separate section that describes specifying the analysis rule.

> **Note**
>
> There is an analysis rule type called the ID mapping table analysis rule. However, this analysis rule is managed by AWS Clean Rooms and can't be modified. For more information, see ID mapping table analysis rule.

The following sections describe supported use cases and controls for each analysis rule type.

Supported use cases

The following tables show a comparison summary of the supported use cases for each analysis rule type.

Use case	Aggregation	List	Custom
Supported analyses	Queries that aggregate statistics using COUNT, SUM, and AVG functions along optional dimensions	Queries that output row-level lists of the overlap between multiple tables	Any custom analysis as long as the analysis template or the analysis creator have been reviewed and allowed
Common use cases	Segment analysis, measurement, attribution	Enrichment, segment building	First-touch attributi on, incremental analyses, audience discovery
SQL constructs	• JOIN statements: INNER JOIN • Aggregate functions: COUNT/ COUNT DISTINCT, SUM/SUM DISTINCT, and AVG • Scalar functions: Limited subset	• JOIN statements: INNER JOIN • Scalar functions: None	• Majority of SQL functions and SQL constructs available with the SELECT command • A restricted SQL surface applies when minimum aggregation thresholds are

Use case	Aggregation	List	Custom
			configured. For more information, see SQL capabilities for minimum aggregation and comparison controls.
Subqueries and common table expressions (CTEs)	No	No	Yes
Analysis templates	No	No	Yes

Supported controls

The following tables show a comparison summary of how each analysis rule type protects your underlying data.

Control	Aggregation	List	Custom
Control mechanism	Control how data in the table can be used in a query *(For example, allow COUNT and SUM of column hashed_email.)*	Control how data in the table can be used in a query *(For example, allow use of column hashed_email only for joining.)*	Control what queries are allowed to run on the table *(For example, allow only queries defined in analysis templates "Custom query 1".)*
Built-in privacy enhancing techniques	• Blind match • Aggregation required • Min aggregation threshold >=	• Blind match • Overlap required • Pre-defined query structure	• Minimum aggregation thresholds • Comparison controls

Control	Aggregation	List	Custom
	• 2 Pre-defined query structure	• Allowed additional analyses	• Disallowed output columns • Differential privacy
Review query before it can be run	No	No	Yes, using analysis templates

For more information about the analysis rules that are available in AWS Clean Rooms, see the following topics.

- Aggregation analysis rule
- List analysis rule
- Custom analysis rule in AWS Clean Rooms

Aggregation analysis rule

In AWS Clean Rooms, an *aggregation analysis rule* generates aggregate statistics using COUNT, SUM, and/or AVG functions along optional dimensions. When the aggregation analysis rule is added to a configured table, it enables the member who can query to run queries on the configured table.

The aggregation analysis rule supports uses cases such as campaign planning, media reach, frequency measurement, and attribution.

The supported query structure and syntax are defined in Aggregation query structure and syntax.

The parameters of the analysis rule, defined in Aggregation analysis rule - query controls, include query controls and query results controls. Its query controls include the ability to require that a configured table is joined to at least one configured table owned by the member who can query, either directly or transitively. This requirement allows you to ensure that the query is run on the intersection (INNER JOIN) of your table and theirs.

Aggregation query structure and syntax

Queries on tables that have an aggregation analysis rule must adhere to the following syntax.

```
--select_aggregate_function_expression
SELECT
aggregation_function(column_name) [[AS] column_alias ] [, ...]

 --select_grouping_column_expression
  [, {column_name|scalar_function(arguments)} [[AS] column_alias ]][, ...]

--table_expression
FROM table_name [[AS] table_alias ]
  [[INNER] JOIN table_name [[AS] table_alias] ON join_condition] [...]

--where_expression
[WHERE where_condition]

--group_by_expression
[GROUP BY {column_name|scalar_function(arguments)}, ...]]

--having_expression
[HAVING having_condition]

--order_by_expression
[ORDER BY {column_name|scalar_function(arguments)} [{ASC|DESC}]] [,...]]
```

The following table explains each expression listed in the preceding syntax.

Expression	Definition	Examples
select_aggregate_function_expression	A comma-separated list containing the following expressions: • `select_aggregation_function_expression` • `select_aggregate_expression`	`SELECT SUM(PRICE), user_segment`

Expression	Definition	Examples
	Note There must be at least one `select_aggregation_function_expression` in the `select_aggregate_expression`.	
`select_aggregation_function_expression`	One or more supported aggregation functions applied to one or more columns. Only columns are allowed as arguments of aggregation functions. **Note** There must be at least one `select_aggregation_function_expression` in the `select_aggregate_expression`.	`AVG(PRICE)` `COUNT(DISTINCT user_id)`

Expression	Definition	Examples

select_grouping_column_expression	An expression that can contain any expression using the following:
	• Table column names
	• Supported scalar functions
	• String literals
	• Numerical literals
	
	Note
	`select_aggregate_expression` can alias columns with or without the AS parameter. For more information, see the AWS Clean Rooms SQL Reference.
		`TRUNC(timestampColumn)`
		
		`UPPER(campaignName)`

Expression	Definition	Examples
table_expression	A table, or join of tables, connecting join conditional expressions with `join_condition` . `join_condition` returns a Boolean. The `table_expression` supports: • A specific JOIN type (INNER JOIN) • The equality comparison condition within a `join_condition` (=) • Logical operators (AND, OR).	`FROM consumer_table INNER JOIN provider_table ON consumer_table.identifier1 = provider_table.identifier1 AND consumer_table.identifier2 = provider_table.identifier2`

Expression	Definition	Examples

where_expression	A conditional expression that returns a Boolean. It may be comprised of the following:
	• Table column names
	• Supported scalar functions
	• Mathematical operators
	• String literals
	• Numerical literals
	
	Supported comparison conditions are (`=, >, <, <=, >=, <>, !=, NOT, IN, NOT IN, LIKE, IS NULL, IS NOT NULL`).
	
	Supported logical operators are (`AND, OR`).
	
	The `where_expression` is optional.
		`WHERE where_condition`
		
		`WHERE price > 100`
		
		`WHERE TRUNC(timestampColumn) = '1/1/2022'`
		
		`WHERE timestampColumn = timestampColumn2 - 14`

group_by_expression	A comma-separated list of expressions that match the requirements for the `select_grouping_column_expression` .	`GROUP BY TRUNC(timestampColumn), UPPER(campaignName), segment`

Expression	Definition	Examples
having_expression	A conditional expression that returns an Boolean. They have a supported aggregati on function applied to a single column (for example, `SUM(price)`) and are compared to a numerical literal. Supported conditions are (`=`, `>`, `<`, `<=`, `>=`, `<>`, `!=`). Supported logical operators are (`AND`, `OR`). The `having_expression` is optional.	`HAVING SUM(SALES) > 500`

Expression	Definition	Examples
order_by_expression	A comma-separated list of expressions that is compatible with the same requirements defined in `select_aggregate_expression` defined earlier. The `order_by_expression` is optional. **Note** `order_by_expression` permits ASC and DESC parameters. For more information, see ASC DESC parameters in the AWS Clean Rooms SQL Reference.	`ORDER BY SUM(SALES), UPPER(campaignName)`

For aggregation query structure and syntax, be aware of the following:

- SQL commands other than SELECT are not supported.
- Sub-queries and common table expressions (for example, WITH) are not supported.
- Operators that combine multiple queries (for example, UNION) are not supported.
- TOP, LIMIT, and OFFSET parameters are not supported.

Aggregation analysis rule - query controls

With aggregation query controls, you can control how the columns in your table are used to query the table. For example, you can control which column is used for joining, which column can be counted, or which column can be used in WHERE statements.

The following sections explain each control.

Topics

- Aggregation controls
- Join controls
- Dimension controls
- Scalar functions

Aggregation controls

By using *aggregation controls*, you can define which aggregation functions to allow, and what columns they must to be applied to. Aggregation functions can be used in the SELECT, HAVING, and ORDER BY expressions.

Control	Definition	Usage
`aggregateColumns`	Columns of configured table columns you allow for use within aggregation functions.	`aggregateColumns` can be used inside an aggregati on function in the SELECT, HAVING, and ORDER BY expressions. Some `aggregateColumns` can also be categorized as a `joinColumn` (defined later). Given `aggregateColumn` can't also be categorized as a `dimensionColumn` (defined later).
`function`	The COUNT, SUM, and AVG functions you allow for use on top of the `aggregate Columns` .	`function` can be applied to an `aggregateColumns` that is associated to it.

Join controls

A `JOIN` clause is used to combine rows from two or more tables, based on a related column between them.

You can use *Join controls* to control how your table can be joined to other tables in the `table_expression`. AWS Clean Rooms only supports INNER JOIN. INNER JOIN statements can only use columns that have been explicitly categorized as a `joinColumn` in your analysis rule, subject to the controls that you define.

The INNER JOIN must operate on a `joinColumn` from your configured table and a `joinColumn` from another configured table in the collaboration. You decide which columns from your table can be used as `joinColumn`.

Each match condition within the ON clause is required to use the equality comparison condition (=) between two columns.

Multiple match conditions within an ON clauses can be:

- Combined using the `AND` logical operator
- Separated using the `OR` logical operator

> **Note**
>
> All JOIN match conditions must match one row from each side of the JOIN. All conditionals connected by an `OR` or an `AND` logical operator must adhere to this requirement as well.

The following is an example of a query with an `AND` logical operator.

```
SELECT some_col, other_col
FROM table1
    JOIN table2
    ON table1.id = table2.id AND table1.name = table2.name
```

The following is an example of a query with an `OR` logical operator.

```
SELECT some_col, other_col
FROM table1
    JOIN table2
```

```
    ON table1.id = table2.id OR table1.name = table2.name
```

Control	Definition	Usage
`joinColumns`	The columns (if any) that you want to allow the member who can query to use in the INNER JOIN statement.	A specific `joinColumn` can also be categorized as a `aggregateColumn` (see Aggregation controls). The same column can't be used both as `joinColumn` and `dimensionColumns` (see later). Unless it has also been categorized as an `aggregateColumn`, a `joinColumn` can't be used in any other parts of the query other than the INNER JOIN.
`joinRequired`	Control whether you require an INNER JOIN with a configured table from the member who can query.	If you enable this parameter, an INNER JOIN is required. If you don't enable this parameter, an INNER JOIN is optional. Assuming you enable this parameter, the member who can query is required to include a table they own in the INNER JOIN. They must JOIN your table with theirs, either directly or transitively (that is, join their table to another table, which itself is joined to your table).

Following is an example of transitivity.

```
ON
my_table.identifer = third_party_table.identifier
....
ON
third_party_table.identifier = member_who_can_query_table.id
```

> **Note**
>
> The member who can query can also use the `joinRequired` parameter. In that case, the query must join their table with at least one other table.

Dimension controls

Dimension controls control the column along which the aggregation columns can be filtered, grouped, or aggregated.

Control	Definition	Usage
`dimensionColumns`	The columns (if any) that you allow the member who can query to use in SELECT, WHERE, GROUP BY, and ORDER BY.	A `dimensionColumn` can be used in SELECT (`select_grouping_column_expression`), WHERE, GROUP BY, and ORDER BY. The same column can't be both a `dimensionColumn`, a `joinColumn`, and/or an `aggregateColumn`.

Scalar functions

Scalar functions control which scalar functions can be used on dimension columns.

Control	Definition	Usage
`scalarFunctions`	The scalar functions that can be used on `dimensionColumns` in the query.	Specifies the scalar functions (if any) that you allow (for example, CAST) to be applied on `dimensionColumns` . Scalar functions can't be used on top of other functions or within other functions. Arguments of scalar functions can be columns, string literals, or numeric literals.

The following scalar functions are supported:

- Math functions – ABS, CEILING, FLOOR, LOG, LN, ROUND, SQRT
- Data type formatting functions – CAST, CONVERT, TO_CHAR, TO_DATE, TO_NUMBER, TO_TIMESTAMP
- String functions – LOWER, UPPER, TRIM, RTRIM, SUBSTRING
  - For RTRIM, custom character sets to trim aren't allowed.
- Conditional expressions – COALESCE
- Date functions – EXTRACT, GETDATE, CURRENT_DATE, DATEADD
- Other functions – TRUNC

For more details, see the AWS Clean Rooms SQL Reference.

Aggregation analysis rule - query results controls

With the aggregation query results controls, you can control which results are returned by specifying one or more conditions that each output row must meet for it to be returned. AWS Clean Rooms supports aggregation constraints in the form of `COUNT (DISTINCT column) >= X`. This form requires that each row aggregates at least X distinct values of a choice from your configured table (for example, a minimum number of distinct `user_id` values). This minimum threshold is automatically enforced, even if the submitted query itself does not use the specified

column. They are enforced collectively across each configured table in the query from the configured tables from each member in the collaboration.

Each configured table must have at least one aggregation constraint in their analysis rule. Configured table owners can add multiple `columnName` and associated `minimum` and they are enforced collectively.

Aggregation constraints

Aggregation constraints control which rows in the query results are returned. To be returned, a row must meet the specified minimum number of distinct values in each column specified in the aggregation constraint. This requirement applies even if the column isn't explicitly mentioned in the query or in other parts of the analysis rule.

Control	Definition	Usage
`columnName`	The `aggregateColumn` that is used in the condition that each output row must meet.	Can be any column in the configured table.
`minimum`	The minimum number of distinct values for the associated `aggregate Column` that the output row must have (for example, COUNT DISTINCT) for it to be returned in the query results.	The `minimum` must be at least value of 2.

Aggregation analysis rule structure

The following example shows a predefined structure for an aggregation analysis rule.

In the following example, *`MyTable`* refers to your data table. You can replace each *`user input placeholder`* with your own information.

```
{
  "aggregateColumns": [
    {
      "columnNames": [MyTable column names], "function": [Allowed Agg Functions]
```

```
    },
  ],
  "joinRequired": ["QUERY_RUNNER"],
  "joinColumns": [MyTable column names],
  "dimensionColumns": [MyTable column names],
  "scalarFunctions": [Allowed Scalar functions],
  "outputConstraints": [
    {
      "columnName": [MyTable column names], "minimum": [Numeric value]
    },
  ]
}
```

Aggregation analysis rule - example

The following example demonstrates how two companies can collaborate in AWS Clean Rooms using aggregation analysis.

Company A has customer and sales data. Company A is interested in understanding product return activity. Company B is one of Company A's retailers and has returns data. Company B also has segment attributes on customers that are useful to Company A (for example, purchased related products, uses customer service from the retailer). Company B doesn't want to provide row-level customer return data and attribute information. Company B only wants to enable a set of queries for Company A to obtain aggregate statistics on overlapping customers at a minimum aggregation threshold.

Company A and Company B decide to collaborate so that Company A can understand product return activity and deliver better products at Company B and other channels.

To create the collaboration and run an aggregation analysis, the companies do the following:

1. Company A creates a collaboration and creates a membership. The collaboration has Company B as another member in the collaboration. Company A enables query logging in the collaboration, and it enables query logging in their account.
2. Company B creates a membership in the collaboration. It enables query logging in its account.
3. Company A creates a sales configured table.
4. Company A adds the following aggregation analysis rule to the sales configured table.

```
{
  "aggregateColumns": [
```

```
    {
      "columnNames": [
        "identifier"
      ],
      "function": "COUNT_DISTINCT"
    },
    {
      "columnNames": [
        "purchases"
      ],
      "function": "AVG"
    },
    {
      "columnNames": [
        "purchases"
      ],
      "function": "SUM"
    }
  ],
  "joinColumns": [
    "hashedemail"
  ],
  "dimensionColumns": [
    "demoseg",
    "purchasedate",
    "productline"
  ],
  "scalarFunctions": [
    "CAST",
    "COALESCE",
    "TRUNC"
  ],
  "outputConstraints": [
    {
      "columnName": "hashedemail",
      "minimum": 2,
      "type": "COUNT_DISTINCT"
    },
  ]
}
```

`aggregateColumns` – Company A wants to count the number of unique customers in the overlap between sales data and returns data. Company A also wants to sum the number of `purchases` made to compare to number of `returns`.

`joinColumns` – Company A wants to use `identifier` to match customers from sales data to customers from returns data. This will help Company A match returns to the right purchases. It also helps Company A segment overlapping customers.

`dimensionColumns` – Company A uses `dimensionColumns` to filter by the specific product, compare purchases and returns over a certain period of time, ensure the return date is after the product date, and help segment overlapping customers.

`scalarFunctions` – Company A selects `CAST` scalar function to help update data type formats if needed based on the configured table Company A associates to the collaboration. It also adds scalar functions to help formatting columns if needed.

`outputConstraints` – Company A sets minimum output constraints. It doesn't need to constrain the results because the analyst is allowed to see row-level data from their sales table

> **Note**
>
> Company A doesn't include `joinRequired` in the analysis rule. It provides flexibility for their analyst to query the sales table alone.

5. Company B creates a returns configured table.
6. Company B adds the following aggregation analysis rule to the returns configured table.

```
{
  "aggregateColumns": [
    {
      "columnNames": [
        "identifier"
      ],
      "function": "COUNT_DISTINCT"
    },
    {
      "columnNames": [
        "returns"
      ],
```

```
      "function": "AVG"
    },
    {
      "columnNames": [
        "returns"
      ],
      "function": "SUM"
    }
  ],
  "joinColumns": [
    "hashedemail"
  ],
  "joinRequired": [
    "QUERY_RUNNER"
  ],
  "dimensionColumns": [
    "state",
    "popularpurchases",
    "customerserviceuser",
    "productline",
    "returndate"
  ],
  "scalarFunctions": [
    "CAST",
    "LOWER",
    "UPPER",
    "TRUNC"
  ],
  "outputConstraints": [
    {
      "columnName": "hashedemail",
      "minimum": 100,
      "type": "COUNT_DISTINCT"
    },
    {
      "columnName": "producttype",
      "minimum": 2,
      "type": "COUNT_DISTINCT"
    }
  ]
}
```

`aggregateColumns` – Company B enables Company A to sum `returns` to compare to the number of purchases. They have at least one aggregate column because they are enabling an aggregate query.

`joinColumns` – Company B enables Company A to join on `identifier` to match customers from return data to customers from sales data. `identifier` data is particularly sensitive and having it as a `joinColumn` ensures that the data will never be outputted in a query.

`joinRequired` – Company B requires queries on the return data to be overlapped with the sales data. They don't want to enable Company A to query all individuals in their dataset. They also agreed on that restriction in their collaboration agreement.

`dimensionColumns` – Company B enables Company A to filter and group by `state`, `popularpurchases`, and `customerserviceuser` which are unique attributes that could help make the analysis for Company A. Company B enables Company A to use `returndate` to filter output on `returndate` that occurs after `purchasedate`. With this filtering, the output is more accurate for evaluating the impact of the product change.

`scalarFunctions` – Company B enables the following:

- TRUNC for dates
- LOWER and UPPER in case the `producttype` is entered in a different format in their data
- CAST if Company A needs to convert data types in sales to be the same as data types in returns

Company A doesn't enable other scalar functions because they don't believe they are required for queries.

`outputConstraints` – Company B sets minimum output constraints on `hashedemail` to help reduce the ability to re-identify customers. It also adds minimum output constraint on `producttype` to reduce the ability to re-identify specific products that were returned. Certain product types could be more dominant based on dimensions of the output (for example, `state`). Their output constraints will always be enforced regardless of output constraints added by Company A to their data.

7. Company A creates a sales table association to collaboration.
8. Company B creates a returns table association to collaboration.

9. Company A runs queries, such as the following example, to better understand the quantity of returns in Company B as compared to total purchases by location in 2022.

```
SELECT
  companyB.state,
  SUM(companyB.returns),
  COUNT(DISTINCT companyA.hashedemail)
FROM
  sales companyA
  INNER JOIN returns companyB ON companyA.identifier = companyB.identifier
WHERE
  companyA.purchasedate BETWEEN '2022-01-01' AND '2022-12-31' AND
  TRUNC(companyB.returndate) > companyA.purchasedate
GROUP BY
  companyB.state;
```

10. Company A and Company B review query logs. Company B verifies that the query aligns with what was agreed upon in the collaboration agreement.

Troubleshooting aggregation analysis rule issues

Use the information here to help you diagnose and fix common issues when you work with aggregation analysis rules.

Issues

- My query didn't return any results

My query didn't return any results

This can happen when there are no matching results or when the matching results don't meet one or more minimum aggregation thresholds.

For more information about minimum aggregation thresholds, see Aggregation analysis rule - example.

List analysis rule

In AWS Clean Rooms, a *list analysis rule* outputs row-level lists of the overlap between the configured table that it's added to and the configured tables of the member who can query. The member who can query runs queries that include a list analysis rule.

The list analysis rule type supports uses cases such as enrichment and audience building.

For more information about the predefined query structure and syntax for this analysis rule, see List analysis rule predefined structure.

The parameters of the list analysis rule, defined in List analysis rule - query controls, have query controls. Its query controls include the ability to select the columns that can be listed in the output. The query is required to have at least one join with a configured table from the member who can query, either directly or transitively.

There are no query results controls like there are for the Aggregation analysis rule.

List queries can only use mathematical operators. They can't use other functions (such as aggregation or scalar).

Topics

- List query structure and syntax
- List analysis rule - query controls
- List analysis rule predefined structure
- List analysis rule - example

List query structure and syntax

Queries on tables that have a list analysis rule must adhere to the following syntax.

```
--select_list_expression
SELECT DISTINCT column_name [[AS] column_alias ] [, ...]

--table_expression
FROM table_name [[AS] table_alias ]
  [[INNER] JOIN table_name [[AS] table_alias] ON join_condition] [...]

--where_expression
[WHERE where_condition]

--limit_expression
[LIMIT number]
```

The following table explains each expression listed in the preceding syntax.

Expression	Definition	Examples
select_list_expression	A comma-separated list containing at least one table column name. A `DISTINCT` parameter is required. **Note** The `select_list_expression` can alias columns with or without the `AS` parameter. For more information, see the AWS Clean Rooms SQL Reference.	`SELECT DISTINCT segment`
table_expression	A table, or join of tables, with `join_condition` to connect it to `join_condition`. `join_condition` returns a Boolean. The `table_expression` supports: • A specific JOIN type (INNER JOIN) • The equality comparison conditions within a `join_condition` (=)	`FROM consumer_table INNER JOIN provider_table ON consumer_table.identifier1 = provider_table.identifier1 AND consumer_table.identifier2 = provider_table.identifier2`

Expression	Definition	Examples
	• Logical operators (AND, OR).	

where_expression	A conditional expression that returns a Boolean. It can be comprised of the following:
	• Table column names
	• Mathematical operators
	• String literals
	• Numerical literals
	
	Supported comparison conditions are (`=, >, <, <=, >=, <>, !=, NOT, IN, NOT IN, LIKE, IS NULL, IS NOT NULL`).
	
	Supported logical operators are (`AND, OR`).
	
	The `where_expression` is optional.
		`WHERE state + '_' + city = 'NY_NYC'`
		
		`WHERE timestampColumn = timestampColumn2 - 14`

limit_expression	This expression must take a positive integer. The `limit_expression` is optional.	`LIMIT 100`

For list query structure and syntax, be aware of the following:

- SQL commands other than SELECT are not supported.
- Subqueries and common table expressions (for example, WITH) are not supported
- HAVING, GROUP BY, and ORDER BY clauses are not supported
- OFFSET parameter is not supported

List analysis rule - query controls

With list query controls, you can control how the columns in your table are used to query the table. For example, you can control which column is used for joining, or which column can be used in SELECT statement and WHERE clause.

The following sections explain each control.

Topics

- Join controls
- List controls

Join controls

With *Join controls*, you can control how your table can be joined to other tables in the **table_expression**. AWS Clean Rooms only supports INNER JOIN. In the list analysis rule, at least one INNER JOIN is required and the member who can query is required to include a table they own in the INNER JOIN. This means they must join your table with theirs, either directly or transitively.

Following is an example of transitivity.

```
ON
my_table.identifer = third_party_table.identifier
....
ON
third_party_table.identifier = member_who_can_query_table.id
```

INNER JOIN statements can only use columns that have been explicitly categorized as a `joinColumn` in your analysis rule.

The INNER JOIN must operate on a `joinColumn` from your configured table and a `joinColumn` from another configured table in the collaboration. You decide which columns from your table can be used as `joinColumn`.

Each match condition within the ON clause is required to use the equality comparison condition (=) between two columns.

Multiple match conditions within an ON clause can be:

- Combined using the `AND` logical operator

- Separated using the OR logical operator

> **Note**
>
> All JOIN match conditions must match one row from each side of the JOIN. All conditionals connected by an OR or an AND logical operator must adhere to this requirement as well.

The following is an example of a query with an AND logical operator.

```
SELECT some_col, other_col
FROM table1
    JOIN table2
    ON table1.id = table2.id AND table1.name = table2.name
```

The following is an example of a query with an OR logical operator.

```
SELECT some_col, other_col
FROM table1
    JOIN table2
    ON table1.id = table2.id OR table1.name = table2.name
```

Control	Definition	Usage
`joinColumns`	The columns that you want to allow the member who can query to use in the INNER JOIN statement.	The same column can't be categorized as both a `joinColumn` and `listColumn` (see List controls). `joinColumn` can't be used in any other parts of the query other than INNER JOIN.

List controls

List controls control the columns that can be listed in the query output (that is, used in the SELECT statement) or used to filter results (that is, used in the WHERE statement).

Control	Definition	Usage
`listColumns`	The columns that you allow the member who can query to use in the SELECT and WHERE	A `listColumn` can be used in SELECT and WHERE. The same column can't be used as both a `listColumn` and `joinColumn` .

List analysis rule predefined structure

The following example includes a predefined structure that shows how you complete a list analysis rule.

In the following example, *`MyTable`* refers to your data table. You can replace each *`user input placeholder`* with your own information.

```
{
  "joinColumns": [MyTable column name(s)],
  "listColumns": [MyTable column name(s)],
}
```

List analysis rule - example

The following example demonstrates how two companies can collaborate in AWS Clean Rooms using list analysis.

Company A has customer relationship management (CRM) data. Company A wants to obtain additional segment data on its customers to learn more about their customers and potentially use attributes as input into other analyses. Company B has segment data comprised of unique segment attributes that they created based on their first party data. Company B wants to provide the unique segment attributes to Company A only on customers that are overlapping between their data and Company A data.

The companies decide to collaborate so that Company A can enrich the overlapping data. Company A is the member who can query, and Company B is the contributor.

To create a collaboration and run list analysis in collaboration, the companies do the following:

1. Company A creates a collaboration and creates a membership. The collaboration has Company B as another member on the collaboration. Company A enables query logging in the collaboration, and it enables query logging in its account.
2. Company B creates a membership in the collaboration. It enables query logging in its account.
3. Company A creates a CRM configured table
4. Company A adds the analysis rule to the customer configured table, as shown in the following example.

```
{
  "joinColumns": [
    "identifier1",
    "identifier2"
  ],
  "listColumns": [
    "internalid",
    "segment1",
    "segment2",
    "customercategory"
  ]
}
```

   `joinColumns` – Company A wants to use `hashedemail` and/or `thirdpartyid` (obtained from an identity vendor) to match customers from CRM data to customers from segment data. This will help ensure Company A matches enriched data for the right customers. They have two joinColumns to potentially improve the match rate of the analysis.

   `listColumns` – Company A uses `listColumns` to obtain enriched columns beside an `internalid` they use within their own systems. They add `segment1`, `segment2`, and `customercategory` to potentially limit the enrichment to specific segments by using them in filters.
5. Company B creates a segment configured table.
6. Company B adds the analysis rule to the segment configured table.

```
{
  "joinColumns": [
    "identifier2"
  ],
  "listColumns": [
    "segment3",
```

```
    "segment4"
  ]
}
```

`joinColumns` – Company B enables Company A to join on `identifier2` to match customers from segment data to CRM data. Company A and Company B worked with the identity vendor to obtain `identifier2` which would match for this collaboration. They didn't add other `joinColumns` because they believed `identifier2` provides the highest and most accurate match rate and other identifiers aren't required for the queries.

`listColumns` – Company B enables Company A to enrich their data with `segment3` and `segment4` attributes which are unique attributes they have created, collected and aligned on (with customer A) to be a part of data enrichment. They want Company A to obtain these segments for the overlap at a row-level because this is a data enrichment collaboration.

7. Company A creates a CRM table association to the collaboration.
8. Company B creates a segment table association to the collaboration.
9. Company A runs queries, such as the following one to enrich overlapping customer data.

```
SELECT companyA.internalid, companyB.segment3, companyB.segment4
INNER JOIN returns companyB
 ON companyA.identifier2 = companyB.identifier2
WHERE companyA.customercategory > 'xxx'
```

10. Company A and Company B review query logs. Company B verifies that the query aligns with what was agreed upon in the collaboration agreement.

Custom analysis rule in AWS Clean Rooms

In AWS Clean Rooms, a *custom analysis rule* is a new type of analysis rule that allows custom queries to be run on the configured table. Custom SQL queries are still restricted to having only the SELECT command but can use more SQL constructs than aggregation and list queries (for example, window functions, OUTER JOIN, CTEs, or subqueries; see the AWS Clean Rooms SQL Reference for a complete list). Custom SQL queries don’t have to follow a query structure like aggregation and list queries.

The custom analysis rule supports more advanced use cases than those that can be supported by the aggregation and list analysis rule such as custom attribution analysis, benchmarking,

incrementality analysis, and audience discovery. This is in addition to a superset of the use cases supported by aggregation and list analysis rule.

The custom analysis rule also supports differential privacy. Differential privacy is a mathematically-rigorous framework for data privacy protection. For more information, see AWS Clean Rooms Differential Privacy. When you create an analysis template, AWS Clean Rooms Differential Privacy checks the template to determine whether it is compatible with the general-purpose query structure for AWS Clean Rooms Differential Privacy. This validation ensures that you don't create an analysis template that isn't allowed with a differential privacy protected table.

To configure the custom analysis rule, data owners can choose to allow specific custom queries, stored in analysis templates, to run on their configured tables. Data owners review analysis templates before adding them to the allowed analysis control in the custom analysis rule. Analysis templates are available and visible only in the collaboration in which they are created (even if the table is associated to other collaborations) and can be run only by the member who can query in that collaboration.

Alternatively, members can choose to allow other members (query providers) to create queries without review. Members add query providers' accounts the allowed query providers control in the custom analysis rule. If the query provider is the member who can query, they could run any query directly on the configured table. Query providers could also create queries by creating analysis templates. Any queries that have been created by the query providers are automatically allowed to run on the table in all collaborations in which the AWS account is present and the table is associated.

This page contains the following sections:

- Custom analysis rule structure
- Custom analysis rule example with analysis templates
- Custom analysis rule example with minimum aggregation thresholds
- Custom analysis rule example with minimum aggregation thresholds and comparison controls
- Putting it all together

The custom analysis rule supports the following privacy-enhancing controls:

- Allowed analyses
- Disallowed output columns

- Minimum aggregation thresholds
- Comparison controls
- Custom analysis rule with differential privacy
- Considerations and limitations
- SQL capabilities for minimum aggregation and comparison controls

Custom analysis rule structure

The following predefined structure shows the available controls in a custom analysis rule. Include only the controls you need for your use case. The `userIdentifier` value in the `differentialPrivacy` control is the column that uniquely identifies your users, such as *user_id*. When you have two or more tables with differential privacy turned on in a collaboration, AWS Clean Rooms requires you to configure the same column as the user identifier column in both analysis rules. This maintains a consistent definition of the users across tables.

```
{
  "allowedAnalyses": ["ANY_QUERY"] | string[],
  "allowedAnalysisProviders": [],
  "disallowedOutputColumns": [],
  "aggregationThresholds": [
    {
      "identityColumns": [],
      "minimumIdentityCount": number,
      "type": "COUNT_DISTINCT",
      "allowedAggregateExpressionType": "COLUMNS_ONLY" | "ANY_EXPRESSION",
      "outputColumnThresholds": [
        {
          "outputColumnName": string,
          "minimumIdentityCount": number
        }
      ]
    }
  ],
  "comparisonControls": {
    "allowedLiteralComparisonColumns": [],
    "allowedColumnComparisonColumns": []
  },
  "differentialPrivacy": {
    "columns": [
      {
```

```
        "name": "userIdentifier"
      }
    ]
  }
}
```

You can either:

- Add analysis template ARNs to allowed analyses control. In this case, the `allowedAnalysisProviders` control is not included.

```
{
  allowedAnalyses: string[]
}
```

- Add member AWS account IDs to the `allowedAnalysisProviders` control. In this case, you add `ANY_QUERY` to the `allowedAnalyses` control.

```
{
  allowedAnalyses: ["ANY_QUERY"],
  allowedAnalysisProviders: string[]
}
```

You can also configure any of the following controls:

- The columns you do not allow to be projected in the query result. For more information, see Disallowed output columns.

```
{
  disallowedOutputColumns: string[]
}
```

- A minimum aggregation threshold that requires each result row to represent at least a minimum number of distinct data subjects. You can override the threshold for individual output columns through `outputColumnThresholds`, and `allowedAggregateExpressionType` controls whether expressions are permitted inside aggregate functions. For more information, see Minimum aggregation thresholds, Overriding the minimum aggregation threshold for specific output columns, and Allowing nested expressions in aggregate functions.

```
{
```

```
  aggregationThresholds: [
    {
      identityColumns: string[],
      minimumIdentityCount: number,
      type: "COUNT_DISTINCT",
      allowedAggregateExpressionType: "COLUMNS_ONLY" | "ANY_EXPRESSION",
      outputColumnThresholds: [
        {
          outputColumnName: string,
          minimumIdentityCount: number
        }
      ]
    }
  ]
}
```

- Comparison controls that define which columns can be compared to a literal value and which can be compared to another column. For more information, see Comparison controls.

```
{
  comparisonControls: {
    allowedLiteralComparisonColumns: string[],
    allowedColumnComparisonColumns: string[]
  }
}
```

- A differential privacy configuration that protects the table by identifying the user identifier column. For more information, see AWS Clean Rooms Differential Privacy.

```
{
  differentialPrivacy: {
    columns: [
      {
        name: string
      }
    ]
  }
}
```

Configuring minimum aggregation thresholds together with comparison controls is the recommended configuration. A threshold on its own still leaves low-cardinality or quasi-identifying columns comparable, which can narrow results in unintended ways.

Comparison controls are worth configuring when your table contains low-cardinality or quasi-identifying columns — such as postal code or age band — or when the query runner is not fully trusted. For a worked example that shows both controls configured together, see Custom analysis rule example with minimum aggregation thresholds and comparison controls.

Custom analysis rule example with analysis templates

The following example demonstrates how two companies can collaborate in AWS Clean Rooms using the custom analysis rule.

Company A has customer and sales data. Company A is interested in understanding the sales incrementality of an advertising campaign on Company B site. Company B has viewership data and segment attributes that are useful to Company (for example, the device they used when viewing the advertising).

Company A has a specific incrementality query they want to run in the collaboration.

To create a collaboration and run a custom analysis in collaboration, the companies do the following:

1. Company A creates a collaboration and creates a membership. The collaboration has Company B as another member on the collaboration. Company A enables query logging in the collaboration, and it enables query logging in its account.
2. Company B creates a membership in the collaboration. It enables query logging in its account.
3. Company A creates a CRM configured table
4. Company A adds empty custom analysis rule to sales configured table.
5. Company A associates sales configured table to the collaboration.
6. Company B creates viewership configured table.
7. Company B adds an empty custom analysis rule to the viewership configured table.
8. Company B associates viewership configured table to the collaboration.
9. Company A views the sales table and viewership table associated to the collaboration and creates analysis template, adding the incrementality query and parameter for campaign month.

```
{
    "analysisParameters": [
    {
        "defaultValue": ""
        "type": "DATE"
        "name": "campaign_month"
    }
    ],
    "description": "Monthly incrementality query using sales and viewership data"
    "format": "SQL"
    "name": "Incrementality analysis"
    "source":
        "WITH labeleddata AS
        (
        SELECT hashedemail, deviceid, purchases, unitprice, purchasedate,
        CASE
            WHEN testvalue IN ('value1', 'value2', 'value3') THEN 0
            ELSE 1
        END AS testgroup
        FROM viewershipdata
        )
        SELECT labeleddata.purchases, provider.impressions
        FROM labeleddata
        INNER JOIN salesdata
          ON labeleddata.hashedemail = provider.hashedemail
        WHERE MONTH(labeleddata.purchasedate) > :campaignmonth
        AND testgroup = :group
       "
}
```

10. Company A adds their account (for example, 444455556666) to the allowed analysis provider control in the custom analysis rule. They use the allowed analysis provider control because they want to allow any queries they create to run on their sales configured table.

```
{
  "allowedAnalyses": [
    "ANY_QUERY"
  ],
  "allowedAnalysisProviders": [
    "444455556666"
  ]
```

```
}
```

11. Company B sees the created analysis template in the collaboration and reviews its contents including the query string and parameter.
12. Company B determines that the analysis template achieves the incrementality use case and meets their privacy requirements for how their viewership configured table can be queried.
13. Company B adds the analysis template ARN to the allowed analysis control in the custom analysis rule of the viewership table. They use the allowed analysis control because they only want to allow the incrementality query to run on their viewership configured table.

```
{
  "allowedAnalyses": [
    "arn:aws:cleanrooms:us-east-1:111122223333:membership/41327cc4-bbf0-43f1-b70c-
a160dddceb08/analysistemplate/1ff1bf9d-781c-418d-a6ac-2b80c09d6292"
  ]
}
```

14. Company A runs the analysis template and uses the parameter value `05-01-2023`.

Custom analysis rule example with minimum aggregation thresholds

The following example demonstrates how two companies can collaborate in AWS Clean Rooms using the custom analysis rule with minimum aggregation thresholds instead of reviewing individual analysis templates.

Company A is a publisher with an `impressions` table containing `user_id`, `campaign_id`, and `event_date`. Company B is an advertiser that wants to measure campaign reach — the number of distinct users who saw a given campaign. Company A wants to make sure no query result can reveal individuals or small groups, so it uses minimum aggregation thresholds rather than reviewing individual analysis templates.

To create a collaboration and run a custom analysis, the companies do the following:

1. Company A creates a collaboration with Company B as another member and as the member who can query. Company A enables query logging in the collaboration and in its account.
2. Company B creates a membership in the collaboration and enables query logging in its account.
3. Company A creates an `impressions` configured table.

4. Company A adds a custom analysis rule to the `impressions` configured table with a minimum aggregation threshold so that every returned row represents at least 100 distinct users. Company A sets `user_id` as the identity column and overrides the threshold to 5 for the lower-sensitivity `campaign_id` output column. Company A also allows literal comparison on `campaign_id` and `event_date` so that Company B can scope a query to one campaign and a date range. Every column that appears in a literal comparison must be in the allowlist. Finally, Company A adds Company B's account to the allowed analysis providers control so Company B can run queries without per-template review.

```
{
  "aggregationThresholds": [
    {
      "identityColumns": [
        "user_id"
      ],
      "minimumIdentityCount": 100,
      "type": "COUNT_DISTINCT",
      "allowedAggregateExpressionType": "COLUMNS_ONLY",
      "outputColumnThresholds": [
        {
          "outputColumnName": "campaign_id",
          "minimumIdentityCount": 5
        }
      ]
    }
  ],
  "comparisonControls": {
    "allowedLiteralComparisonColumns": [
      "campaign_id",
      "event_date"
    ]
  },
  "allowedAnalyses": [
    "ANY_QUERY"
  ],
  "allowedAnalysisProviders": [
    "444455556666"
  ]
}
```

5. Company A associates the `impressions` configured table to the collaboration.
6. Company B runs a reach query grouped by `event_date` and filtered to a campaign:

```
SELECT event_date, COUNT(DISTINCT user_id) AS reach
FROM impressions
WHERE event_date >= '2026-01-01' AND campaign_id = 'Holiday Promotion'
GROUP BY event_date;
```

7. AWS Clean Rooms returns only the rows backed by at least 100 distinct users and suppresses the rest, so Company B learns daily reach for the Holiday Promotion campaign without learning about any individual or small group.

event_date	reach
2026-01-01	142
2026-01-02	118
2026-01-04	103

The date `2026-01-03` does not appear in the results because fewer than 100 distinct users saw the campaign that day, so AWS Clean Rooms suppressed that row.

The key difference from the analysis-templates approach is that Company A never reviewed a specific query. Instead, Company A relies on the threshold to constrain what any query can return. For more information, see Minimum aggregation thresholds and Comparison controls.

Custom analysis rule example with minimum aggregation thresholds and comparison controls

Configuring minimum aggregation thresholds together with comparison controls is the recommended baseline configuration. A threshold alone ensures that every result row represents a minimum number of distinct data subjects, but it does not prevent comparisons on low-cardinality or quasi-identifying columns. Without comparison controls, a query runner can still filter or join on those columns, potentially narrowing results in unintended ways.

Comparison controls are worth configuring when your table contains low-cardinality or quasi-identifying columns — such as postal code or age band — or when the query runner is not fully trusted. Adding comparison controls restricts which columns can appear in literal comparisons and column-to-column comparisons, closing the gap that a threshold alone leaves open.

The following configuration combines both controls:

```
{
  "aggregationThresholds": [
    {
      "identityColumns": [
        "user_id"
      ],
      "minimumIdentityCount": 100,
      "type": "COUNT_DISTINCT",
      "allowedAggregateExpressionType": "COLUMNS_ONLY"
    }
  ],
  "comparisonControls": {
    "allowedLiteralComparisonColumns": [
      "campaign_id"
    ],
    "allowedColumnComparisonColumns": [
      "user_id"
    ]
  },
  "allowedAnalyses": [
    "ANY_QUERY"
  ],
  "allowedAnalysisProviders": [
    "444455556666"
  ]
}
```

With this configuration, every result row represents at least 100 distinct data subjects. The query runner can filter by `campaign_id` using literal comparisons and join on `user_id` using column-to-column comparisons. Because the comparison allowlists are set, any column not listed — including low-cardinality columns such as postal code or age band — cannot be used in a comparison at all.

For more information about each control, see Minimum aggregation thresholds and Comparison controls. For a fuller configuration that also includes disallowed output columns, see Putting it all together.

Putting it all together

The following example shows a complete configuration of the custom analysis rule type, using disallowed output columns, minimum aggregation thresholds, and comparison controls. This

configuration enforces minimum aggregation of 100 distinct data subjects, prevents `user_id` from being projected in the query result, and allows the query runner to analyze the intersection of customers joining on the `user_id` column.

This policy also grants additional flexibility by allowing literal comparison filtering on low-sensitivity columns `status` and `price`, and overrides the minimum aggregation threshold to 5 for the `campaign_id` column:

```
{
  "disallowedOutputColumns": [
    "user_id"
  ],
  "comparisonControls": {
    "allowedLiteralComparisonColumns": [
      "status",
      "price"
    ],
    "allowedColumnComparisonColumns": [
      "user_id"
    ]
  },
  "aggregationThresholds": [
    {
      "identityColumns": [
        "user_id"
      ],
      "minimumIdentityCount": 100,
      "type": "COUNT_DISTINCT",
      "allowedAggregateExpressionType": "COLUMNS_ONLY",
      "outputColumnThresholds": [
        {
          "outputColumnName": "campaign_id",
          "minimumIdentityCount": 5
        }
      ]
    }
  ],
  "allowedAnalyses": [
    "ANY_QUERY"
  ],
  "allowedAnalysisProviders": [
    "444455556666",
    "333366669999"
```

```
  ]
}
```

Allowed analyses

The `allowedAnalyses` control in the custom analysis rule lets you allow specific custom queries, stored in analysis templates, to run on your configured table. You review analysis templates before adding them to this control. For more information about analysis templates, see Analysis templates in AWS Clean Rooms.

Analysis templates are collaboration-specific and visible only in the collaboration where they are created. Only the member who can query in that collaboration can run analysis templates.

You can allow analysis templates or accounts to create queries, not both. If you leave the `allowedAnalyses` control empty, the member who can query cannot run queries on the configured table.

To allow any query without reviewing individual templates, set `allowedAnalyses` to `["ANY_QUERY"]` and use the `allowedAnalysisProviders` control together. For more information, see Allowed analysis providers.

The following example adds an analysis template ARN to the `allowedAnalyses` control:

```
{
  "allowedAnalyses": [
    "arn:aws:cleanrooms:us-east-1:111122223333:membership/41327cc4-bbf0-43f1-b70c-
a160dddceb08/analysistemplate/1ff1bf9d-781c-418d-a6ac-2b80c09d6292"
  ]
}
```

Allowed analysis providers

Instead of reviewing individual analysis templates, you can allow other members (query providers) to create queries without review. You add the query providers' AWS account IDs to the `allowedAnalysisProviders` control. When you use this control, set `allowedAnalyses` to `["ANY_QUERY"]`.

If the query provider is the member who can query, they can run any query directly on the configured table. Query providers can also create analysis templates.

AWS Clean Rooms automatically allows any query that an allowed query provider creates to run on the table in all collaborations where that AWS account is present and the table is associated.

> **Scope of allowed analysis providers**
>
> Allowing analysis providers is more permissive than reviewing templates individually. When you add a query provider, all current and future queries from that AWS account are allowed to run on your configured table. For finer control, use `allowedAnalyses` with specific analysis template ARNs instead.

The following example adds a query provider AWS account ID to the `allowedAnalysisProviders` control:

```
{
  "allowedAnalyses": [
    "ANY_QUERY"
  ],
  "allowedAnalysisProviders": [
    "444455556666"
  ]
}
```

Minimum aggregation thresholds with analysis templates

AWS Clean Rooms enforces minimum aggregation thresholds on queries that run from analysis templates, not only on queries submitted directly by allowed analysis providers. If you configure a minimum aggregation threshold on your configured table and also allow specific analysis templates, AWS Clean Rooms still suppresses any result row that does not meet the threshold.

This provides an additional guardrail. Reviewing an analysis template tells you what a query is allowed to do. The minimum aggregation threshold independently constrains what any allowed query can return. The two controls work together, so you do not have to rely on template review alone to prevent results about individuals or small groups.

For more information, see Minimum aggregation thresholds.

Disallowed output columns

The `disallowedOutputColumns` control in the custom analysis rule lets you define the list of columns that cannot be projected in the query result. Any reference to a disallowed column

through transformation, aliasing, or other means cannot appear in the final SELECT (projection) of the query.

The control prohibits direct projection but does not fully prevent values from being indirectly inferred. You can still use disallowed columns in a projection inside a subquery or common table expression (CTE), as long as they are not referenced in the final projection.

> **CACHE TABLE constraint**
>
> AWS Clean Rooms enforces the disallowed output columns constraint on cached tables. A cached table cannot reference a disallowed output column in its SELECT clause. To use a column with a disallowed output column constraint in a subsequent part of your query, convert the cached table to a CTE.

The following example adds `user_id` to the `disallowedOutputColumns` control:

```
{
  "disallowedOutputColumns": [
    "user_id"
  ]
}
```

For more information about how disallowed columns work with configured tables, see Configured table disallowed columns.

Minimum aggregation thresholds

Data aggregation is a privacy-enhancing technique that helps enforce anonymization by preventing queries from returning results about individuals or small groups. When minimum aggregation thresholds are configured on your tables, AWS Clean Rooms suppresses results that do not meet the data provider's specified minimum threshold.

Topics

- Aggregation thresholds
- Overriding the minimum aggregation threshold for specific output columns
- Allowing nested expressions in aggregate functions
- Configuring minimum aggregation thresholds in the console
- Considerations and limitations

Aggregation thresholds

You can configure minimum aggregation thresholds in a Custom analysis rule for your configured table. Data providers specify the `identityColumns` and the `minimumIdentityCount`, N. Supported values for `minimumIdentityCount` are between 2 and 100,000. The `identityColumns` value must be a `string`, `varchar`, or `char` type. Minimum aggregation thresholds ensure that each row in a SQL query result has at least N distinct data subjects contributing to it. For information about how `identityColumns` interacts with query filters, see Comparison controls.

The following example sets the minimum aggregation threshold to 100 with `user_id` as the `identityColumns` value:

```
{
  "aggregationThresholds": [
    {
      "identityColumns": [
        "user_id"
      ],
      "minimumIdentityCount": 100,
      "type": "COUNT_DISTINCT"
    }
  ]
}
```

Overriding the minimum aggregation threshold for specific output columns

Output columns have different sensitivity levels, and data providers can enforce different `minimumIdentityCount` values depending on the output column. For example, `campaign_id` might be a low-sensitivity column with a minimum threshold of 5 distinct data subjects, while `postal_code` and `income_bracket` get a minimum threshold of 100 because they are high-sensitivity columns.

A per-column override accepts `0`, which exempts that output column from minimum aggregation, or a value between 2 and 100,000.

The following example sets a minimum aggregation threshold override for the `campaign_id` column:

```
{
```

```
  "outputColumnThresholds": [
    {
      "outputColumnName": "campaign_id",
      "minimumIdentityCount": 5
    }
  ]
}
```

Allowing nested expressions in aggregate functions

Allowing expressions inside aggregation functions increases the risk that data subjects can be re-identified in clean room queries. The default setting for `allowedAggregateExpressionType` is `COLUMNS_ONLY`, which is recommended as the privacy-enhanced configuration. With this setting, only columns are accepted inside an aggregation function, such as:

- SUM(cost)
- COUNT(DISTINCT campaign_id)

Alternatively, for collaborations where the query runner is a trusted partner, you can allow `ANY_EXPRESSION` inside an aggregation function. This setting increases the flexibility of SQL queries that can run on your data, but introduces privacy risk because query runners can isolate small groups within an aggregation function.

> **Privacy and compliance review required**
>
> Before allowing `ANY_EXPRESSION` inside aggregate functions, consult your privacy, security, legal, and compliance teams to make sure that this adheres to your organization's requirements.

The following is an example of an expression in an aggregation function that is generally considered safe:

- SUM(cost * quantity)

The following is an example of an expression in an aggregation function that is considered a privacy risk:

- SUM(CASE WHEN zip_code = '10001' THEN salary ELSE 0 END)

The following is a complete `aggregationThresholds` example:

```
{
  "aggregationThresholds": [
    {
      "identityColumns": [
        "user_id"
      ],
      "minimumIdentityCount": 100,
      "type": "COUNT_DISTINCT",
      "allowedAggregateExpressionType": "COLUMNS_ONLY",
      "outputColumnThresholds": [
        {
          "outputColumnName": "campaign_id",
          "minimumIdentityCount": 5
        }
      ]
    }
  ]
}
```

Configuring minimum aggregation thresholds in the console

You configure minimum aggregation thresholds as part of adding a custom analysis rule to a configured table, after the configured table exists. For the complete custom analysis rule guided flow, see Adding a custom analysis rule to a table (guided flow).

To configure minimum aggregation thresholds

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**, and then choose your configured table.
3. On the configured table detail page, choose **Configure analysis rule**.
4. For **Analysis rule type**, choose **Custom**.
5. Choose either **Review each new analysis** or **Allow any analysis by specific collaborators**. The remaining steps follow the **Allow any analysis by specific collaborators** path.
6. Choose **Add AWS account** to configure which partners can submit analyses on this configured table, then choose the AWS account to allow.

7. On **Specify analysis results controls - optional**, for **Enforce aggregation thresholds?**, choose **Yes**.
8. Configure the minimum aggregation threshold.
   a. Choose your identity column. The identity column must be a `string`, `varchar`, or `char` type.
   b. Specify your minimum identity count. Supported values are between 2 and 100,000.
   c. **Aggregate type** is preconfigured to **Count distinct**.
9. For **Allow nested expressions in aggregate functions?**, the default is **No**. Leaving this option turned off is the privacy-enhanced configuration. For more information, see Allowing nested expressions in aggregate functions.
10. (Optional) Configure output column overrides.
    a. Choose **Add column override**.
    b. Choose your output column from the dropdown list.
    c. Specify your minimum identity count. Supported values are `0` to exempt the column from minimum aggregation, or between 2 and 100,000.

    For more information, see Overriding the minimum aggregation threshold for specific output columns.
11. (Recommended) Configure columns allowed for comparison.
    a. By default, all columns are eligible for comparison. Explicitly configuring which columns are allowed for literal and column-to-column comparison is the privacy-enhanced configuration.
    b. Choose **Custom list**.
    c. Choose which columns are allowed for literal comparison. For example, allowing the `state` column lets the query runner run a query with a literal filter, such as `WHERE state = 'New York'`.
    d. Choose which columns can be used for column-to-column comparison. For example, allowing the `state` column lets the query runner run a cross-table join with a column comparison, such as `JOIN my_table t ON t.state = partner_table.state`.

    For more information, see Comparison controls.

12. Choose **Next**.
13. Choose **Next**. Differential privacy is turned off and is not supported with minimum aggregation thresholds or comparison controls. For more information, see Limitations.
14. Review your custom analysis rule configuration.
15. Choose **Configure analysis rule**.
16. Associate the configured table to a collaboration. For more information, see Step 2: Associate a configured table.

Considerations and limitations

Considerations and limitations apply when you use minimum aggregation thresholds. For the full list, see Considerations and limitations.

Comparison controls

In SQL, literal comparison matches a column from your dataset against a literal value typed directly into the query. Column comparison matches values from two different columns against each other, either within the same table or across joined tables.

Default behavior

Comparison controls are an allowlist. Once you set `allowedLiteralComparisonColumns`, only the columns you list can be compared to a literal value, and every column you do not list is blocked. The same applies to `allowedColumnComparisonColumns` for column-to-column comparisons. Adding a column to one allowlist does not add it to the other.

If you do not configure `comparisonControls`, AWS Clean Rooms applies no comparison restrictions — a query can compare any column to a literal value or to another column.

If you do not configure `comparisonControls` but you do set a minimum aggregation threshold, comparisons remain otherwise unrestricted. However, AWS Clean Rooms never allows a literal comparison on a column listed in `identityColumns`. That restriction comes from the threshold itself, so it applies whether or not you configure comparison controls. For more information, see Minimum aggregation thresholds.

If you configure `comparisonControls` together with disallowed output columns, the two controls are independent and both apply. Comparison controls govern which columns a query can compare; disallowed output columns govern which columns can appear in the query result. A

column can be allowed in a comparison and still be barred from the result. For more information, see Disallowed output columns.

Literal comparison

A literal comparison evaluates a column against a single hardcoded constant (a string, number, or date). The right side of the operator never changes during the query execution. Syntax examples include:

- WHERE status = 'Active'
- WHERE price > 49.99

When you use Minimum aggregation thresholds, AWS Clean Rooms does not allow literal comparison on the `identityColumns` value. This prevents the query runner from submitting a query filtered down to an individual or specific set of users. Avoid allowing literal comparison on low cardinality columns that can single out small groups or individual data subjects.

```
{
  "comparisonControls": {
    "allowedLiteralComparisonColumns": [
      "status",
      "price"
    ]
  }
}
```

Choosing columns for literal comparison

Allow literal comparison only on columns that don't identify individuals or small groups. Avoid allowing it on low-cardinality columns (for example, age_band, coarse region code). Even though these columns are not the configured `identityColumns` value, comparing them to literals can narrow results to a small, identifiable population. High-cardinality, non-identifying dimensions such as `campaign_id` or `product_sku` are safer choices.

Example: Allowing literal comparison on a campaign column

A publisher configures a custom analysis rule with a minimum aggregation threshold so that every output row represents at least 100 distinct users (`user_id`). An advertiser runs queries against

this table but needs to scope their analysis to a specific advertising campaign — for example, to measure reach for one campaign at a time.

Because `user_id` is the identity column, it can't be compared to a literal, preventing the advertiser from filtering results down to a single user. But `campaign_id` and `event_date` are high-cardinality, non-identifying dimensions, so the publisher adds them to `allowedLiteralComparisonColumns`, which lets the advertiser filter by campaign and scope the analysis to a date range:

```
{
  "aggregationThresholds": [
    {
      "identityColumns": ["user_id"],
      "minimumIdentityCount": 100
    }
  ],
  "comparisonControls": {
    "allowedLiteralComparisonColumns": ["campaign_id", "event_date"]
  }
}
```

Given the preceding configuration, this query is allowed:

```
-- Allowed: campaign_id and event_date are both in allowedLiteralComparisonColumns
SELECT campaign_id, COUNT(DISTINCT user_id) AS reach
FROM impressions
WHERE campaign_id = 'CMP-1024'
  AND event_date >= '2026-01-01'
GROUP BY campaign_id;
```

Given the same configuration, this query is blocked:

```
-- Blocked: user_id is the identity column and can never be compared to a literal
SELECT campaign_id, COUNT(DISTINCT user_id) AS reach
FROM impressions
WHERE user_id = 'U-88231'
GROUP BY campaign_id;
```

The first query still returns only rows backed by at least 100 distinct users, while the literal comparisons on `campaign_id` and `event_date` filter which rows are considered. The second query is rejected because it attempts to single out an individual data subject.

Column comparison

A column comparison evaluates the value of one column against the value of another column dynamically for every single row. Syntax examples include:

- WHERE retail_price < wholesale_price
- WHERE users.id = orders.user_id

When using Minimum aggregation thresholds, a data provider can allow column comparison on the `identityColumns` value for use cases that require joining across tables, such as an audience overlap report.

```
{
  "comparisonControls": {
    "allowedColumnComparisonColumns": [
      "user_id"
    ]
  }
}
```

Example: Allowing column comparison on the identity column for an overlap report

A publisher and an advertiser want to measure their audience overlap — how many users appear in both of their datasets — without either party learning who any individual user is. This requires joining the two tables on `user_id`, which is a column-to-column comparison. The publisher allows the advertiser to run an audience overlap analysis scoped to specific campaign IDs.

Because `user_id` is the identity column, the publisher has already blocked literal comparisons on it (so no one can filter to a specific person). To enable the join across tables, the publisher adds `user_id` to `allowedColumnComparisonColumns`. To enable campaign filtering, the publisher adds `campaign_id` to `allowedLiteralComparisonColumns`.

```
{
  "aggregationThresholds": [
    {
      "identityColumns": ["user_id"],
      "minimumIdentityCount": 100
    }
  ],
  "comparisonControls": {
```

```
    "allowedLiteralComparisonColumns": ["campaign_id"],
    "allowedColumnComparisonColumns": ["user_id"]
  }
}
```

Given the preceding configuration, this query is allowed:

```
-- Allowed: user_id is compared against another column (column-to-column join)
SELECT COUNT(DISTINCT p.user_id) AS overlapping_users
FROM publisher_audience p
  JOIN advertiser_audience a
ON p.user_id = a.user_id;
```

Given the same configuration, this query is blocked:

```
-- Blocked: email is not in allowedColumnComparisonColumns
SELECT COUNT(DISTINCT p.user_id) AS overlapping_users
FROM publisher_audience p
  JOIN advertiser_audience a
  ON p.email = a.email;
```

The join succeeds because a column-to-column comparison evaluates dynamically per row and doesn't let the query runner target a known value. The result enforces the minimum aggregation threshold, ensuring the overlap count is only returned if it represents at least 100 distinct users. The second query is blocked because `email` is not in the `allowedColumnComparisonColumns` allowlist and cannot be used in a comparison — only `user_id` can.

Comparison controls and expressions

Comparison controls follow indirect literal comparisons, not only a direct WHERE `column = 'literal'` predicate. If you allow `ANY_EXPRESSION` inside aggregate functions through `allowedAggregateExpressionType`, comparison controls still block a literal comparison on a column that is not in `allowedLiteralComparisonColumns`. This applies even when the literal is nested inside an expression.

The following example shows a query that remains blocked because `zip_code` is not in the allowlist, even though the literal is inside a CASE expression rather than written as a direct predicate:

```
-- Blocked: zip_code is not in allowedLiteralComparisonColumns,
```

```
-- even though the literal comparison is nested inside a CASE expression
SELECT SUM(CASE WHEN zip_code = '00001' THEN salary ELSE 0 END) AS total
FROM employees;
```

This is why the two controls are complementary: allowing expressions inside aggregates widens what a query can compute, while comparison controls still constrain which columns a query can single out by value. For more information, see Allowing nested expressions in aggregate functions.

Custom analysis rule with differential privacy

In AWS Clean Rooms, the custom analysis rule supports differential privacy. Differential privacy is a mathematically-rigorous framework for data privacy protection that helps you protect your data against re-identification attempts.

Differential privacy supports aggregate analysis such as ad campaign planning, post-ad-campaign measurement, benchmarking in a financial institution consortium, and A/B testing for healthcare research.

The supported query structure and syntax are defined in Query structure and syntax.

Custom analysis rule with differential privacy example

> **Note**
>
> AWS Clean Rooms Differential Privacy is only available for collaborations where the data is stored in Amazon S3.

Consider the custom analysis rule example presented in the previous section. This example demonstrates how you can use differential privacy to protect your data against re-identification attempts while allowing your partner to learn business-critical insights from your data. Assume that Company B, who has the viewership data, wants to protect their data using differential privacy. To complete the differential privacy setup, Company B completes the following steps:

1. Company B turns on differential privacy while adding custom analysis rule to the viewership configured table. Company B selects `viewershipdata.hashedemail` as the user identifier column.
2. Company B adds a differential privacy policy in the collaboration to make their viewership data table available for querying. Company B selects the default policy to quickly complete the setup.

Company A, who wants to understand the sales incrementality of an advertising campaign on Company B's site, runs the analysis template. Because the query is compatible with the general-purpose query structure of AWS Clean Rooms Differential Privacy, the query runs successfully.

Query structure and syntax

Queries containing at least one table that have the differential privacy turned on must adhere to the following syntax.

```
query_statement:
    [cte, ...] final_select

 cte:
    WITH sub_query AS (
       inner_select
       [ UNION | INTERSECT | UNION_ALL | EXCEPT/MINUS ]
       [ inner_select ]
    )

 inner_select:
     SELECT [user_id_column, ] expression [, ...]
     FROM table_reference [, ...]
     [ WHERE condition ]
     [ GROUP BY user_id_column[, expression] [, ...] ]
     [ HAVING condition ]

 final_select:
     SELECT [expression, ...] | COUNT | COUNT_DISTINCT | SUM | AVG | STDDEV
     FROM table_reference [, ...]
     [ WHERE condition ]
     [ GROUP BY expression [, ...] ]
     [ HAVING COUNT | COUNT_DISTINCT | SUM | AVG | STDDEV | condition ]
     [ ORDER BY column_list ASC | DESC ]
     [ OFFSET literal ]
     [ LIMIT literal ]

 expression:
    column_name [, ...] | expression AS alias | aggregation_functions |
 window_functions_on_user_id | scalar_function | CASE | column_name math_expression [,
 expression]

 window_functions_on_user_id:
```

```
    function () OVER (PARTITION BY user_id_column, [column_name] [ORDER BY column_list
ASC|DESC])
```

> **Note**
>
> For differential privacy query structure and syntax, be aware of the following:
>
> - Sub-queries are not supported.
> - Common Table Expressions (CTEs) should emit the user identifier column if a table or CTE involve data protected by differential privacy. Filters, groupings, and aggregations should be done at the user level.
> - Final_select allows COUNT DISTINCT, COUNT, SUM, AVG, and STDDEV aggregate functions.

For more details about which SQL keywords are supported for differential privacy, see SQL capabilities of AWS Clean Rooms Differential Privacy.

Considerations and limitations

AWS Clean Rooms helps you and your partners analyze and collaborate on collective datasets to drive new insights without revealing underlying data to one another. Verify that your data collaboration configuration adheres to the privacy, security, legal, and compliance requirements of your organization.

AWS Clean Rooms offers various privacy-enhancing controls for your data collaboration use case:

- Allowed analyses
- Disallowed output columns
- Minimum aggregation thresholds
- Comparison controls
- Differential privacy

Consider comparison controls when your table contains low-cardinality or quasi-identifying columns — such as postal code or age band — or when the query runner is not fully trusted. For more information, see Comparison controls.

Assumptions about potentially adversarial queries

To help you and your partners maintain the controls you need, AWS Clean Rooms makes the following assumptions regarding privacy-enhanced data collaboration:

- Assume the query runner is trying to exfiltrate user level data
- Assume other data providers are trying to exfiltrate user level data
- Assume the query runner and other data providers are colluding to exfiltrate user level data

Limitations

Note the following limitations:

- Data aggregation enforced by minimum aggregation thresholds on its own does not address other potential exfiltration risks such as differencing attacks. For more information, see Differencing attacks.
- The SQL surface for queries using minimum aggregation thresholds is restricted. For a complete list of supported and unsupported SQL constructs, see SQL capabilities for minimum aggregation and comparison controls.
- Minimum aggregation thresholds and comparison controls do not currently work with differential privacy.
- The minimum aggregation threshold is not enforced on COUNT, COUNT(DISTINCT), or APPROX_COUNT_DISTINCT functions over a single table with no grouping or join.

Differencing attacks

A differencing attack occurs when someone runs two queries that differ by a single individual, then subtracts the results to isolate that person's data, even though each query on its own only returns aggregates.

For example:

- Query A: SUM(salary) WHERE department = 'Sales' → 1,000,000
- Query B: SUM(salary) WHERE department = 'Sales' AND name != 'Jane' → 950,000
- Difference: Jane earns 50,000.

Each query passes a minimum-group-size check (minimum aggregation threshold), but the combination leaks one data subject's value by running multiple queries. A per-query threshold does not address this differencing attack vector.

To reduce the risk of bulk attacks, adopt data access budgets and differential privacy policies.

SQL capabilities for minimum aggregation and comparison controls

The following table lists the supported and unsupported SQL constructs and functions when a custom analysis rule uses minimum aggregation thresholds and comparison controls in the Spark analytics engine. Function categories follow the AWS Clean Rooms Spark SQL reference.

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Query constructs	• CACHE TABLE • Query hints • Analysis template (paramete rized queries)	Yes	Supported	Supported
Clauses	• SELECT • SELECT DISTINCT • FROM • FROM SUBQUERY • FROM INLINE TABLE (VALUES) • WHERE • GROUP BY • HAVING • ORDER BY	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• WITH (CTE)			
	• GROUP BY CUBE • GROUP BY ROLLUP • GROUP BY GROUPING SETS • EXISTS • CLUSTER BY • DISTRIBUTE BY • SORT BY • LATERAL JOIN • PIVOT / UNPIVOT • WITH RECURSIVE	No	—	—
Join clauses	• JOIN INNER • JOIN LEFT • JOIN FULL OUTER • JOIN CROSS • JOIN RIGHT • JOIN NATURAL	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• JOIN LEFT SEMI • JOIN LEFT ANTI	No	—	—
Set operators	• UNION • UNION ALL	Yes	Supported	Supported
	• EXCEPT • EXCEPT ALL • INTERSECT • INTERSECT ALL	No	—	—
Sort and row limits	• ORDER BY ASC / DESC • ORDER BY NULLS FIRST • ORDER BY NULLS LAST • LIMIT	Yes	Supported	Supported
	• OFFSET	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Conditions	• = • != / <> • < • <= • > • >= • IN • NOT IN • LIKE • NOT LIKE • RLIKE • BETWEEN • IS TRUE / IS FALSE • AND • OR • NOT • IS DISTINCT FROM	Yes	Supported	Supported
	• <=> (null-safe equality)	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Aggregate functions	• AVG function • COUNT function • COUNT DISTINCT function • SUM function • MIN function • MAX function • STDDEV / STDDEV_SAMP functions • STDDEV_POP function • VAR_SAMP / VARIANCE functions • VAR_POP function • SKEWNESS function • APPROX_COUNT_DISTINCT function • COLLECT_LIST function • COLLECT_SET function • PERCENTILE function	Yes	Supported	When an aggregation threshold is set: MIN, MAX, PERCENTILE, APPROX_PERCENTILE, MEDIAN, COLLECT_LIST, and COLLECT_SET must be combined with a qualifying aggregation (AVG, COUNT, SUM). They cannot be used as a standalone aggregation. When only comparison controls are set: no such restriction. All other aggregate functions are supported.

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• APPROX_PERCENTILE function • MEDIAN function			
	• ANY_VALUE function • BOOL_AND function • BOOL_OR function	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Array functions	• ARRAY function • ARRAY_CONTAINS function • ARRAY_DISTINCT function • ARRAY_INTERSECT function • ARRAY_JOIN function • ARRAY_REMOVE function • ARRAY_SORT function • ARRAY_UNION function • FLATTEN function • EXPLODE function	Yes	Supported	Supported
	• ARRAY_EXCEPT function	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Conditional expressions	• IS NULL function • IS NOT NULL function • GREATEST function • LEAST function • CASE / WHEN functions • IF function • COALESCE function • NULLIF function • NVL function • NVL2 function	Yes	Supported	Supported
Constructor functions	• STRUCT function • NAMED_STRUCT function • MAP function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Data type formatting functions	• STR_TO_MAP function • BASE64 function • UNBASE64 function • HEX function • UNHEX function • TO_DATE function • DECODE function • ENCODE function • CAST function	Yes	Supported	Supported
	• TO_CHAR function • TO_NUMBER function	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Date and time functions	• CURRENT_DATE function • CURRENT_TIMESTAMP function • TO_TIMESTAMP function • DATE_DIFF function • DATE_TRUNC function • DATE_PART function • EXTRACT function • CONVERT_TIMEZONE function • FROM_UTC_TIMESTAMP function • TIMESTAMP function • DAY / DAYOFMONTH functions • DAYOFWEEK function • WEEKOFYEAR function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• MONTH function • YEAR function • HOUR function • MINUTE function • SECOND function			
	• DAYOFYEAR function • DATE_ADD function • ADD_MONTHS function	No	—	—
Hash functions	• MD5 function • SHA function • SHA1 function • SHA2 function • XXHASH64 function	Yes	Supported	Supported
JSON functions	• GET_JSON_OBJECT function • TO_JSON function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Math functions	• ABS function • CEIL / CEILING functions • FLOOR function • ROUND function • LN function • LOG / LOG10 functions • POWER function • SQRT function • RAND / RANDOM functions • EXP function • SIGN function • TRUNC function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• TRUNCATE function • CBRT function • PI function • RADIANS function • ACOS function • ASIN function • ATAN function • ATAN2 function • COS function • COT function • SIN function • DEGREES function	No	—	—
Mathematical operator symbols	• + • - • * • / • %	Yes	Supported	Supported
Scalar functions	• CARDINALITY function • SIZE function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
String functions	• CONCAT function • LOWER function • UPPER function • LENGTH / CHAR_LENGTH / CHARACTER_LENGTH functions • SUBSTR / SUBSTRING functions • TRIM function • BTRIM function • LTRIM function • RTRIM function • POSITION function • REPLACE function • REPEAT function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• REGEXP_REPLACE function • REGEXP_SUBSTR function • SPLIT function • FORMAT_STRING function • UUID function • LPAD function • RPAD function • LEFT function • RIGHT function			
	• REVERSE function • TRANSLATE function • REGEXP_COUNT function • REGEXP_INSTR function • SPLIT_PART function	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Privacy-related functions	• consent_tcf_v2_decode function • consent_gpp_v1_decode function	Yes	Supported	Supported

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Window functions	• ROW_NUMBER function • RANK function • DENSE_RANK function • LAG function • LEAD function • Window AVG function • Window COUNT function • Window SUM function • Window MIN function • Window MAX function • Window STDDEV_POP function • Window STDDEV_SAMP function • Window VAR_POP function • Window VAR_SAMP function	Yes	Supported	When an aggregation threshold is set: LAG, LEAD, Window MIN, Window MAX, Window APPROX_PERCENTILE, and Window PERCENTILE must be combined with a qualifying aggregation. They cannot be used as a standalone value. When only comparison controls are set: no such restriction. All other window functions are supported.

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
	• Window SKEWNESS function • Window APPROX_COUNT_DISTINCT function • Window APPROX_PERCENTILE function • Window PERCENTILE function			
	• FIRST / FIRST_VALUE functions • LAST / LAST_VALUE functions • NTH_VALUE function • CUME_DIST function • PERCENT_RANK function • NTILE function	No	—	—

Category	SQL construct	Supported	Common table expressions (CTEs)	Final SELECT clause
Encryption and decryption functions	• AES_ENCRYPT function • AES_DECRYPT function	No	—	—
Hyperloglog functions	• HLL_SKETCH_AGG function • HLL_UNION_AGG function • HLL_SKETCH_ESTIMATE function • HLL_UNION function	No	—	—

ID mapping table analysis rule

In AWS Clean Rooms, an *ID mapping table analysis rule* isn't a standalone analysis rule. This type of analysis rule is managed by AWS Clean Rooms and is used to join disparate identity data to facilitate querying. It's automatically added to ID mapping tables and can't be edited. It inherits the behaviors of the other analysis rules in the collaboration – as long as those analysis rules are homogeneous.

The ID mapping table analysis rule enforces security on an ID mapping table. It restricts a collaboration member from directly selecting or inspecting the non-overlap population between the two members' datasets using the ID mapping table. The ID mapping table analysis rule is used to protect the sensitive data in the ID mapping table when used in queries with other analysis rules implicitly.

With the ID mapping table analysis rule, AWS Clean Rooms enforces an overlap on both sides of the ID mapping table in expanded SQL. This enables you to do the following tasks:

- Use the overlap of the ID mapping table in JOIN statements.

  AWS Clean Rooms allows an INNER, LEFT, or RIGHT join on the ID mapping table if it respects the overlap. To protect sensitive mapping information, the ID mapping table must always be on the "inner" side of any JOIN operation. For example, the following JOIN operations are valid:

  - table LEFT JOIN id_mapping_table
  - id_mapping_table RIGHT JOIN table
  - table INNER JOIN id_mapping_table

  The following JOIN operations are invalid:

  - id_mapping_table LEFT JOIN table
  - table RIGHT JOIN id_mapping_table

  This prevents exposure of mapping records that don't have corresponding matches in your dataset. Allowing such operations could potentially reveal sensitive information about other collaboration members' data mappings.
- Use the mapping table columns in JOIN statements.

  You can’t use the mapping table columns in the following statements: SELECT, WHERE, HAVING, GROUP BY, or ORDER BY (unless protections are modified on the source ID namespace association or the target ID namespace association).
- In expanded SQL, AWS Clean Rooms also supports OUTER JOIN, implicit JOIN, and CROSS JOIN. These joins can't satisfy overlap requirements. Instead, AWS Clean Rooms uses `requireOverlap` to specify which columns have to be joined on.

The supported query structure and syntax are defined in ID mapping table query structure and syntax.

The parameters of the analysis rule, defined in ID mapping table analysis rule query controls, include query controls and query results controls. Its query controls include the ability to require overlap of the ID mapping table in JOIN statements (that is, `requireOverlap`).

Topics

- ID mapping table query structure and syntax
- ID mapping table analysis rule query controls
- ID mapping table analysis rule predefined structure

- ID mapping table analysis rule – example

ID mapping table query structure and syntax

Queries on tables that have an ID mapping table analysis rule must adhere to the following syntax.

```
--select_list_expression
SELECT
provider.data_col, consumer.data_col

--table_expression
FROM provider

JOIN idMappingTable idmt ON provider.id = idmt.sourceId

JOIN consumer ON consumer.id = idmt.targetId
```

Collaboration tables

The following tables represent configured tables that exist in an AWS Clean Rooms collaboration. The **id** column in both the **cr_drivers_license** and **cr_insurance** tables represents a column to match with the ID mapping table.

cr_drivers_license

id	driver_name	state_of_registration
1	Eduard	TX
2	Dana	MA
3	Gweneth	IL

cr_insurance

id	policyholder_email	policy_number

a	eduardo@internal.company.com	17f9d04e-f5be-4426-bdc4-250ed59c6529
b	gwen@internal.company.com	3f0092db-2316-48a8-8d44-09cf8f6e6c64
c	rosa@internal.company.com	d7692e84-3d3c-47b8-b46d-a0d5345f0601

ID mapping table

The following table represents an existing ID mapping table that matches on the **cr_drivers_license** and **cr_insurance** tables. Not all entries will have IDs for both collaboration tables.

cr_drivers_license_id	cr_insurance_id
1	a
2	null
3	b
null	c

The ID mapping table analysis rule only allows for queries to run on the set of overlapping data, which would look as follows:

cr_drivers_license_id	cr_insurance_id	driver_name	state_of_registration	policyholder_email	policy_number
1	a	Eduard	TX	eduardo@internal.company.com	17f9d04e-f5be-4426-bdc4-250ed59c6529

3	b	Gweneth	IL	gwen@internal.company.com	3f0092db-2316-48a8-8d44-09cf8f6e6c64

Example queries

The following examples show valid locations for the ID mapping table joins:

```
-- Single ID mapping table
SELECT
    [ select_items ]FROM
    cr_drivers_license cr_dl
    [ INNER | LEFT ] JOIN cr_identity_mapping_table idmt ON idmt.cr_drivers_license_id
 = cr_dl.id
    [ INNER | RIGHT ] JOIN cr_insurance cr_in            ON idmt.cr_insurance_id
 = cr_in.id
;
-- Single ID mapping table (Subquery)
SELECT
    [ select_items ]FROM (
    SELECT
        [ select_items ]
    FROM
        cr_drivers_license cr_dl
        [ INNER | LEFT ] JOIN cr_identity_mapping_table idmt ON
 idmt.cr_drivers_license_id = cr_dl.id
        [ INNER | RIGHT ] JOIN cr_insurance cr_in            ON idmt.cr_insurance_id
    = cr_in.id
)
;
-- Single ID mapping table (CTE)
WITH
    matched_ids AS (
        SELECT
            [ select_items ]
        FROM
            cr_drivers_license cr_dl
            [ INNER | LEFT ] JOIN cr_identity_mapping_table idmt ON
 idmt.cr_drivers_license_id = cr_dl.id
            [ INNER | RIGHT ] JOIN cr_insurance cr_in            ON
 idmt.cr_insurance_id       = cr_in.id
```

```
    )SELECT
    [ select_items ]FROM
    matched_ids
;
```

Considerations

For ID mapping table query structure and syntax, be aware of the following:

- You can't edit it.
- It's applied to the ID mapping table by default.
- It uses a source and target ID namespace association inside the collaboration.
- The ID mapping table is configured by default to provide default protections for the column that comes from the ID namepsace. You can modify this configuration so that the column that comes from the ID namespace (either `sourceID` or `targetID`) can be allowed anywhere in the query. For more information, see ID namespaces in AWS Clean Rooms.
- The ID mapping table analysis rule inherits the SQL restrictions of the other analysis rules in the collaboration.

ID mapping table analysis rule query controls

With ID mapping table query controls, AWS Clean Rooms controls how the columns in your table are used to query the table. For example, it controls which columns are used for joining, and which columns require overlap. The ID mapping table analysis rule also includes functionality that enables you to allow the `sourceID`, the `targetID`, or both, to be projected without requiring a JOIN.

The following table explains each control.

Control	Definition	Usage
`joinColumns`	The columns that the member who can query can use in the INNER JOIN statement.	You can't use `joinColumns` in any other parts of the query other than INNER JOIN. For more information, see Join controls.

Control	Definition	Usage
`dimensionColumns`	The columns (if any) that the member who can query can use in SELECT and GROUP BY statements.	A `dimensionColumn` can be used in SELECT and GROUP BY. A `dimensionColumn` can appear as `joinKeys`. You can only use `dimensionColumns` in the JOIN clause if you specify it in brackets.
`queryContraints:RequireOverlap`	The columns in the ID mapping table that must be joined on so the query can run.	These columns must be used to JOIN the ID Mapping table and a collaboration table.

ID mapping table analysis rule predefined structure

The predefined structure for an ID mapping table analysis rule comes with default protections that are applied to the `sourceID` and `targetID`. This means that the column with protections applied must be used in queries.

You can configure the ID mapping table analysis rule in the following ways:

- Both `sourceID` and `targetID` protected

  In this configuration, the `sourceID` and `targetID` can't both be projected. The `sourceID` and `targetID` must be used in a JOIN when the ID mapping table is referenced.
- Only `targetID` protected

  In this configuration, the `targetID` can't be projected. The `targetID` must be used in a JOIN when the ID mapping table is referenced. The `sourceID` can be used in query.
- Only `sourceID` protected

  In this configuration, the `sourceID` can't be projected. The `sourceID` must be used in a JOIN when ID mapping table is referenced. The `targetID` can be used in query.

- Neither `sourceID` or `targetID` protected

  In this configuration, the ID mapping table isn't subject to any specific enforcement that can be used in query.

The following example shows a predefined structure for an ID mapping table analysis rule with the default protections applied to the `sourceID` and `targetID`. In this example, the ID mapping table analysis rule only allows an INNER JOIN on both the `sourceID` column and the `targetID` column.

```
{
  "joinColumns": [
    "source_id",
    "target_id"
  ],
  "queryConstraints": [
    {
      "requireOverlap": {
        "columns": [
          "source_id",
          "target_id"
        ]
      }
    }
  ],
  "dimensionColumns": [] // columns that can be used in SELECT and JOIN
}
```

The following example shows a predefined structure for an ID mapping table analysis rule with protections applied to the `targetID`. In this example, the ID mapping table analysis rule only allows an INNER JOIN on the `sourceID` column.

```
{
  "joinColumns": [
    "source_id",
    "target_id"
  ],
  "queryConstraints": [
    {
      "requireOverlap": {
        "columns": [
          "target_id"
```

```
        ]
      }
    }
  ],
  "dimensionColumns": [
    "source_id"
  ]
}
```

The following example shows a predefined structure for an ID mapping table analysis rule with protections applied to the `sourceID`. In this example, the ID mapping table analysis rule only allows an INNER JOIN on the `targetID` column.

```
{
  "joinColumns": [
    "source_id",
    "target_id"
  ],
  "queryConstraints": [
    {
      "requireOverlap": {
        "columns": [
          "source_id"
        ]
      }
    }
  ],
  "dimensionColumns": [
    "target_id"
  ]
}
```

The following example shows a predefined structure for an ID mapping table analysis rule without protections applied to the `sourceID` or `targetID`. In this example, the ID mapping table analysis rule allows an INNER JOIN on both the `sourceID` column and the `targetID` column.

```
{
  "joinColumns": [
    "source_id",
    "target_id"
  ],
  "queryConstraints": [
```

```
    {
      "requireOverlap": {
        "columns": []
      }
    }
  ],
  "dimensionColumns": [
    "source_id",
    "target_id"
  ]
}
```

ID mapping table analysis rule – example

Rather than writing a long waterfall statement that references Personally Identifiable Information (PII), for example, companies can use the ID mapping table analysis rule to use multi-party LiveRamp transcoding. The following example demonstrates how you can collaborate in AWS Clean Rooms using the ID mapping table analysis rule.

Company A is an advertiser who has customer and sales data, which will be used as a source. Company A also performs transcoding on behalf of the parties in the collaboration, and brings the LiveRamp credentials.

Company B is a publisher who has event data, which will be used as a target.

> **Note**
>
> Either Company A or Company B can provide LiveRamp transcoding credentials and perform the transcoding.

To create a collaboration that enables the ID mapping table analysis in collaboration, the companies do the following:

1. Company A creates a collaboration and creates a membership. It adds Company B, who also creates a membership in the collaboration.
2. Company A either associates an existing ID namespace source or creates a new one in AWS Entity Resolution using the AWS Clean Rooms console.

   Company A creates a configured table with their sales data and a column keyed to the `sourceId` in the ID mapping table.

The ID namespace source provides data to transcode.

3. Company B either associates an existing ID namespace target or creates a new one in AWS Entity Resolution using the AWS Clean Rooms console.

   Company B creates a configured table with their event data and a column keyed to the `targetId` in the ID mapping table.

   The ID namespace target doesn't provide data to transcode, only metadata around the LiveRamp configuration.
4. Company A discovers the two ID namespaces associated to the collaboration and creates and populates an ID mapping table.
5. Company A runs a query across the two datasets by joining on the ID mapping table.

```
--- this would be valid for Custom or List
SELECT provider.data_col, consumer.data_col
FROM provider
  JOIN idMappingTable-123123123123-myMappingWFName idmt
     ON provider.id = idmt.sourceId
  JOIN consumer
     ON consumer.id = idmt.targetId
```

AWS Clean Rooms Differential Privacy

AWS Clean Rooms Differential Privacy helps you protect the privacy of your users with a mathematically-backed technique that is implemented with intuitive controls in a few clicks. As a fully managed capability, no prior differential privacy experience is needed to help you prevent the re-identification of your users. AWS Clean Rooms automatically adds a carefully calibrated amount of noise to query results at runtime in order to help protect your individual-level data.

AWS Clean Rooms Differential Privacy supports a wide range of analytical queries and is a good fit for a wide variety of use cases, where a small amount of error in the query results will not compromise the usefulness of your analysis. With it, your partners can generate business-critical insights about advertising campaigns, investment decisions, clinical research, and more, all without requiring any additional setup from your partners.

AWS Clean Rooms Differential Privacy protects against overflow or invalid cast errors that make use of scalar functions or math operator symbols in a malicious manner.

For more information about AWS Clean Rooms Differential Privacy, see the following topics.

Topics

- Differential privacy
- How Differential Privacy in AWS Clean Rooms works
- Differential privacy policy
- SQL capabilities of AWS Clean Rooms Differential Privacy
- Differential Privacy query tips and examples
- Limitations of AWS Clean Rooms Differential Privacy

Differential privacy

Differential privacy allows only aggregated insights and obfuscates the contribution of any individual's data in those insights. Differential privacy protects the collaboration data from the member who can receive results learning about a specific individual. Without differential privacy, the member who can receive results can attempt to infer individual user data by adding or removing records about an individual and observing the difference in query results.

When differential privacy is turned on, a specified amount of noise is added to the query results to obfuscate the contribution of individual users. If the member who can receive results tries to observe the difference in query results after removing records about an individual from their dataset, the variability in the query result helps prevent the identification of the individual's data. AWS Clean Rooms Differential Privacy uses the SampCert sampler, a proven correct sampler implementation developed by AWS.

How Differential Privacy in AWS Clean Rooms works

The workflow to turn on differential privacy in AWS Clean Rooms requires the following additional steps when completing the workflow for AWS Clean Rooms:

1. You turn on differential privacy when adding a custom analysis rule.
2. You configure the differential privacy policy for the collaboration to make your data tables protected with differential privacy available for querying.

After you complete these steps, the member who can query can start running queries on differential privacy protected data. AWS Clean Rooms returns results that comply with the

differential privacy policy. AWS Clean Rooms Differential Privacy tracks the estimated number of remaining queries that you can run, similar to the gas gauge in a car that shows you the car's current fuel level. The number of queries that the member who can query can run is limited by the **Privacy budget** and **Noise added per query** parameters that are set in the Differential privacy policy.

Considerations

When using differential privacy in AWS Clean Rooms, consider the following:

- The member who can receive results can't use differential privacy. They will configure a custom analysis rule with differential privacy turned off for their configured tables.
- The member who can query can't join tables from two or more data providers when both have differential privacy turned on.

Differential privacy policy

The differential privacy policy controls how many aggregation functions the member who can query is allowed to run in a collaboration. The **Privacy budget** defines a common, finite resource that is applied all tables in a collaboration. The **Noise added per query** governs the rate at which the privacy budget is depleted.

A differential privacy policy is required to make your differential privacy protected tables available for querying. This is a one-time step in a collaboration and includes two inputs:

- **Privacy budget** – Quantified in terms of epsilon, the privacy budget controls the level of privacy protection. It is a common, finite resource that is applied for all of your tables protected with differential privacy in the collaboration, because the goal is to preserve the privacy of your users whose information can be present in multiple tables.

  The **Privacy budget** is consumed every time a query is run on your tables. When the privacy budget is fully exhausted, the collaboration member who can query can't run additional queries until it is increased or refreshed. By setting a larger privacy budget, the member who can receive results can reduce their uncertainty about individuals within the data. Choose a privacy budget that balances your collaboration requirements against your privacy needs and after consulting with business decision-makers.

  You can select **Refresh privacy budget monthly** to automatically create a new privacy budget each calendar month, if you plan to regularly bring new data into the collaboration. Choosing

this option allows arbitrary amounts of information to be revealed about rows of the data when repeatedly queried across refreshes. Avoid choosing this if the same rows will be repeatedly queried between privacy budget refreshes.

- **Noise added per query** is measured in terms of the number of users whose contributions you want to obscure. This value governs the rate at which the privacy budget is depleted. A larger noise value reduces the rate at which the privacy budget is depleted, and therefore allows more queries to be run on your data. However, this should be balanced against releasing less accurate data insights. Consider the desired accuracy for collaboration insights when setting this value.

You can use the default differential privacy policy to quickly complete the setup or customize your differential privacy policy as per your use case. AWS Clean Rooms Differential Privacy provides intuitive controls to configure the policy. AWS Clean Rooms Differential Privacy lets you preview the utility in terms of the number of aggregations possible across all queries on your data and estimate how many queries can be run in a data collaboration.

You can use the interactive examples to understand how different values of **Privacy budget** and **Noise added per query** would impact the results for different types of SQL queries. In general, you need to balance your privacy needs against the number of queries you want to permit and the accuracy of those queries. A smaller **Privacy budget** or larger **Noise added per query** can better protect user privacy, but provides less meaningful insights to your collaboration partners.

If you increase the **Privacy budget** while keeping the **Noise added per query** parameter the same, the member who can query can run more aggregations on your tables in the collaboration. You can increase the **Privacy budget** any time during the collaboration. If you decrease the **Privacy budget** while keeping the **Noise added per query** parameter the same, the member who can query can run fewer aggregations. You can't decrease the **Privacy budget** after the member who can query has started analyzing your data.

If you increase the **Noise added per query** while keeping the **Privacy budget** input the same, the member who can query can run more aggregations on your tables in the collaboration. If you decrease the **Noise added per query** while keeping the **Privacy budget** input the same, the member who can query can run fewer aggregations. You can increase or decrease the **Noise added per query** any time during the collaboration.

The differential privacy policy is managed by the privacy budget template API actions.

SQL capabilities of AWS Clean Rooms Differential Privacy

AWS Clean Rooms Differential Privacy uses a general-purpose query structure to support complex SQL queries. Custom analysis templates are validated against this structure to ensure that they can run on tables protected by differential privacy. The following table indicates which functions are supported. See Query structure and syntax for more information.

Category	Supported
Common table expressions (CTEs) in the Spark analytics engine	**SELECT statements supported in the Spark analytics engine**
Aggregation functions	**Supported aggregation functions:** • APPROXIMATE_PERCENTILE function • AVG function • COUNT and COUNT DISTINCT functions • MAX function • MEDIAN function • SUM • STDDEV, STDDEV_SAMP, and STDDEV_POP • VAR_SAMP, VAR_POP, and VARIANCE • ANY_VALUE Additional functions: • COALESCE • CAST • CASE • DATE_TRUNC • DATEDIFF • CONCAT • LOWER • UPPER • SUBSTR • TRIM • LEFT • RIGHT • LENGTH • REPLACE • SPLIT_PART • REGEXP_REPLACE • REGEXP_SUBSTR • REGEXP_COUNT • REGEXP_INSTR • REGEXP_SPLIT • REGEXP_EXTRACT • REGEXP_EXTRACT_ALL • REGEXP_LIKE • REGEXP_SPLIT_TO_ARRAY • CHR • ASCII • LPAD • RPAD • REVERSE • LTRIM • RTRIM • TO_CHAR • TO_DATE • TO_TIMESTAMP • TO_NUMBER • CURRENT_DATE • CURRENT_TIMESTAMP • DATE_ADD • DATE_SUB • DATE_PART • EXTRACT • DAYOFWEEK • DAYOFMONTH • DAYOFYEAR • WEEKOFYEAR • MONTHS_BETWEEN • ADD_MONTHS • LAST_DAY • NEXT_DAY • TRUNC • ABS • CEIL • CEILING • FLOOR • ROUND • TRUNCATE • MOD • POWER • SQRT • EXP • LN • LOG • LOG10 • LOG2 • SIGN • SIN • COS • TAN • ASIN • ACOS • ATAN • ATAN2 • SINH • COSH • TANH • ASINH • ACOSH • ATANH • CBRT • DEGREES • RADIANS • PI • RAND • RANDOM • GREATEST • LEAST • NVL • NVL2 • NULLIFY • ZEROIFNULL • ZEROIFNULL • NULLIFZERO • NULLIFZERO • DECODE • CONVERT • CONVERT_TIMEZONE • TO_JSON • JSON_EXTRACT_PATH_TEXT • JSON_EXTRACT_ARRAY_ELEMENT_TEXT • JSON_ARRAY_LENGTH • JSON_SERIALIZE • JSON_PARSE • JSON_DESERIALIZE • JSON_SERIALIZE_TO_VARBYTE • JSON_DESERIALIZE_FROM_VARBYTE • JSON_SERIALIZE_TO_VARCHAR • JSON_DESERIALIZE_FROM_VARCHAR • JSON_SERIALIZE_TO_SUPER • JSON_DESERIALIZE_FROM_SUPER • JSON_SERIALIZE_TO_BINARY • JSON_DESERIALIZE_FROM_BINARY • JSON_SERIALIZE_TO_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSON • JSON_DESERIALIZE_FROM_BINARY_JSON • JSON_SERIALIZE_TO_BINARY_JSONB • JSON_DESERIALIZE_FROM_BINARY_JSONB • JSON_SERIALIZE_TO_BINARY_JSONB_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_SUPER • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_VARBYTE • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_VARBYTE • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_VARCHAR • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_VARCHAR • JSON_SERIALIZE_TO_BINARY_JSONB_BINARY_SUPER • JSON_DESERIALIZE_FROM_BINARY_JSONB_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BINARY_BIN

Common table expressions (CTEs) supported in the Spark analytics engine

- MIN function
- PERCENTILE_CONT function
- STDDEV_SAMP and STDDEV_POP functions
- SUM and SUM DISTINCT functions
- VAR_SAMP and VAR_POP functions

You should write the SELECT expression in those CTEs using `SELECT userIdentifierColumn...` format.

Common table expressions (CTEs)	**Supported in the Spark analytics engine**
WITH clause, common table expressions (CTEs)	Supported with the condition that query using differential privacy protected tables must result in data with user-level records. You should write the SELECT

Common table expressions (CTEs) supported in the Spark analytics engine	expression in those CTEs using `SELECT userIdentifierColumn...' format.

Subquery expressions (CTEs) supported in the Spark analytics engine

Subquerie s can have any subquery that doesn't reference different ial privacy relations in these construct s. You can have any subquery that reference s different ial

- SELECT
- HAVING
- JOIN
- JOIN condition
- FROM
- WHERE

Common table expressions (CTEs) supported in the Spark analytics engine	privacy relations in a FROM and JOIN clause only.

Common table expressions (CTEs) supported in the Spark analytics engine

- INNER JOIN
- LEFT JOIN
- LEFT SEMI JOIN
- LEFT ANTI JOIN
- RIGHT JOIN
- FULL [JOIN]
- CROSS JOIN

supported clauses

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY
- LIMIT
- OR operator
- AS

supported

- SUM
- COUNT
- AVG
- MIN
- MAX
- STDDEV_SAMP
- STDDEV_POP
- VAR_SAMP
- VAR_POP
- COVAR_SAMP
- COVAR_POP
- CORR
- PERCENTILE_CONT
- PERCENTILE_DISC
- APPROX_PERCENTILE

supported

- CAST
- CASE
- COALESCE
- NULLIF
- NOT
- IN
- BETWEEN
- IS NULL
- IS NOT NULL
- LIKE
- EXISTS

supported

- CURRENT_DATE
- CURRENT_TIMESTAMP
- DATE_ADD
- DATE_SUB
- DATEDIFF
- DATE_TRUNC
- EXTRACT
- TO_DATE
- TO_TIMESTAMP

supported

- CONCAT
- LOWER
- LTRIM
- RTRIM
- SUBSTR
- TRIM
- UPPER

supported

- ABS
- CEIL
- FLOOR
- ROUND
- SQRT
- LN
- LOG10
- LOG2
- EXP
- POWER
- MOD
- RAND

supported

- ROW_NUMBER
- RANK
- DENSE_RANK
- NTILE
- LAG
- LEAD
- FIRST_VALUE
- LAST_VALUE

supported

- ARRAY_AGG
- ARRAY_CONCAT
- ARRAY_LENGTH
- ARRAY_ELEMENT

supported

- MAP_KEYS
- MAP_VALUES

supported

- JSON_EXTRACT
- JSON_EXTRACT_SCALAR

supported

- the condition that only functions that are equi-joins or used identifier columns are supported and are mandatory when querying two

Common table expressions (CTEs)	**Supported in the Spark analytics engine**
	or more tables with differential privacy turned on. Ensure that the mandatory equi-join conditions are correct. Confirm that the table owner has configured

Hadoop Hive SELECT statement supported in the Spark analytics engine	**Common table expressions (CTEs)**
the same user identifier column in all tables so that the definition of a user remains consistent across tables.	
CROSS JOIN functions are	

Common table expressions (CTEs)	**SELECT statement supported in the Spark analytics engine**	not supported when combining two or more relations with differential privacy turned on.
UNION, UNION ALL, INTERSECT, EXCEPT | MINUS (these are synonyms)	**UNION supported**	

Athena SELECT statement supported in the Spark analytics engine

Athena aggregate functions supported with the condition that the user identifier column in the window function's partition clause is required when you query a relation

- AVG window function
- COUNT window function
- CUME_DIST window function
- DENSE_RANK window function
- FIRST_VALUE window function
- LAG window function
- LAST_VALUE

Category

Query

SELECT

expressions

supported in the Spark analytics engine

with differential privacy turned on.

- window function
- LEAD window function
- MAX window functions
- MEDIAN window functions
- MIN window functions
- NTH_VALUE window function
- STDDEV_SAMP and STDDEV_POP window function (STDDEV_SAMP

Common table expressions (CTEs) supported in the Spark analytics engine	
	and STDDEV are synonyms) • SUM window functions • VAR_SAMP and VAR_POP window functions (VAR_SAMP and VARIANCE are synonyms)
Ranking functions	
	• DENSE_RANK window function

Common table expressions (CTEs) supported in the Spark analytics engine

- NTILE window function
- PERCENT_RANK window function
- RANK window function
- ROW_NUMBER window function

Common table expressions (CTEs) supported in the Spark analytics engine

All SELECT statement expressions are supported

CASE condition expressions are supported

- COALESCE expression
- GREATEST and LEAST functions
- NVL and COALESCE functions
- NVL2 function
- NULLIF function

Common table expressions (CTEs)	**SELECT statement supported in the Spark analytics engine**	**Comparison and condition**	**Logical conditions**
Supported	Supported	Supported	Supported

- Logical conditions
- Pattern-matching conditions
- BETWEEN range conditions
- Null condition

Common table expressions (CTEs) supported in the Spark analytics engine

All date-time functions supported in transactions

- Concatenation operator
- ADD_MONTHS functions
- CONVERT_TIMEZONE function
- CURRENT_DATE function
- DATEADD function
- DATEDIFF function
- DATE_PART functions

Common table expressions (CTEs) supported in the Spark analytics engine

- DATE_TRUNC function
- EXTRACT function
- TO_TIMESTAMP function
- Date parts for date or timestamp functions

Common table expressions (CTEs) supported in the Spark analytics engine

All scalar functions supported

- Equality operator
- BTRIM function
- CHAR_LENGTH function
- CHARACTER_LENGTH function
- CONCAT function
- LEFT and RIGHT functions
- LEN function
- LENGTH function
- LOWER function

Hadoop Hive SELECT statement expressions supported in the Spark analytics engine

- LPAD and RPAD functions
- LTRIM function
- POSITION functions
- REGEXP_COUNT function
- REGEXP_INSTR function
- REGEXP_REPLACE function
- REGEXP_SUBSTR function
- REPEAT function
- REPLACE function

Common table expressions (CTEs) supported in the Spark analytics engine

- REVERSE function
- RTRIM function
- SPLIT_PART function
- SUBSTRING function
- TRANSLATE function
- TRIM functions
- UPPER function

Common table expressions (CTEs) supported in the Spark analytics engine

All data type formatting functions

- TO_CHAR function
- TO_DATE function
- TO_NUMBER
- Datetime format strings
- Numeric format strings

Common table expressions (CTEs)	**Supported in the Spark analytics engine**
AWS_ENCRYPT and AWS_DECRYPT functions	**supported** • AWS_DECRYPT • ENCODE • DECODE • MD5 function • SHA1 function • SHA2 function • XX_HASH64
All mathematical symbols	**are supported** +, -, *, /, %, and @

Common table expressions (CTEs) supported in the Spark analytics engine

All ANSI SQL functions supported

- ACOS function
- ASIN function
- ATAN function
- ATAN2 function
- CBRT function
- CEILING (or CEIL) function
- COS function
- COT function
- DEGREES function
- LTRIM function

Common table expressions (CTEs) supported in the Spark analytics engine

- EXP function
- FLOOR function
- LN function
- LOG function
- MOD function
- PI function
- POWER function
- RADIANS function
- RANDOM function
- ROUND function
- SIGN function
- SIN function

Common table expressions (CTEs) supported in the Spark analytics engine

- SQRT functions
- TRUNC function

VARBYTE functions supported

- UNBASE64
- HEX
- HLL_SKETCH_AGG,
- HLL_SKETCH_ESTIMATE
- HLL_UNION
- HLL_UNION_AGG

JSON functions supported

- GET_JSON_OBJECT

Query common table expressions (CTEs)	**Supported in the Spark analytics engine**
ARRAY_CONTAINS	Not supported
ARRAY_DISTINCT	Supported
ARRAY_EXCEPT	Supported
ARRAY_INTERSECT	Supported
ARRAY_JOIN	Supported
ARRAY_REMOVE	Supported
ARRAY_SORT	Supported
ARRAY_UNION	Supported
GROUPING SETS, ROLLUP, CUBE	Not supported

Common table expressions (CTEs)	**Supported in the Spark analytics engine**
ORDER BY	**Supported with the condition that an ORDER BY clause is only supported in a window function's partition clause when querying tables with differential privacy**

Common table expressions (CTEs)	**Supported in the Spark analytics engine**
	turned on.
RIGHT, OFFSET	**Not supported**
	is supported in CTEs using differential privacy protected tables
Table and column aliasing	**Supported**
Nested functions on aggregate functions	**Supported**

Common table expressions (CTEs) supported in the Spark analytics engine	**Supported**
Supported functions within aggregate functions	

Common alternatives for unsupported SQL constructs

Category	SQL construct	Alternative
Window functions	• LISTAGG • PERCENTILE_CONT • PERCENTILE_DISC	You can use the equivalent aggregate function with GROUP BY.
Mathematical operator symbols	• $column ||/ 2 • $column |/ 2 • $column ^ 2	• CBRT • SQRT • POWER($column, 2)
Scalar functions	• SYSDATE • $column::integer • convert(type, $column)	• CURRENT_DATE • CAST $column AS integer • CAST $column AS type
Literals	INTERVAL ‘1 SECOND'	INTERVAL '1' SECOND
Row limiting	TOP n	LIMIT n

Category	SQL construct	Alternative
Join	• USING • NATURAL	ON clause should explicitly contain a join criterion.

Differential Privacy query tips and examples

AWS Clean Rooms Differential Privacy uses a general-purpose query structure to support a wide variety of SQL constructs such as Common Table Expressions (CTEs) for data preparation and commonly used aggregate functions such as `COUNT`, or `SUM`. In order to obfuscate the contribution of any possible user in your data by adding noise to aggregate query results at run-time, AWS Clean Rooms Differential Privacy requires that aggregate functions in the final `SELECT statement` are run on user-level data.

The following example uses two tables named `socialco_impressions` and `socialco_users` from a media publisher who wants to protect data using differential privacy while collaborating with an athletic brand with `athletic_brand_sales` data. The media publisher has configured the `user_id` column as the user identifier column while enabling differential privacy in AWS Clean Rooms. The advertiser does not need differential privacy protection and wants to run a query using CTEs on combined data. Since their CTE uses differential privacy protected tables, the advertiser includes the user identifier column from those protected tables in the list of CTE columns and joins the protected tables on the user identifier column.

```
WITH matches_table AS(
     SELECT si.user_id, si.campaign_id, s.sale_id, s.sale_price
     FROM socialco_impressions si
     JOIN socialco_users su
         ON su.user_id = si.user_id
     JOIN athletic_brand_sales s
         ON s.emailsha256 = su.emailsha256
     WHERE s.timestamp > si.timestamp

UNION ALL

     SELECT si.user_id, si.campaign_id, s.sale_id, s.sale_price
     FROM socialco_impressions si
     JOIN socialco_users su
         ON su.user_id = si.user_id
     JOIN athletic_brand_sales s
```

```
        ON s.phonesha256 = su.phonesha256
    WHERE s.timestamp > si.timestamp
)

SELECT COUNT (DISTINCT user_id) as unique_users
FROM matches_table
GROUP BY campaign_id
ORDER BY COUNT (DISTINCT user_id) DESC
LIMIT 5
```

Similarly, if you want to run window functions on differential privacy protected data tables, you must include the user identifier column in the `PARTITION BY` clause.

```
ROW_NUMBER() OVER (PARTITION BY conversion_id, user_id ORDER BY match_type, match_age)
 AS row
```

Limitations of AWS Clean Rooms Differential Privacy

AWS Clean Rooms Differential Privacy doesn't address the following situations:

1. AWS Clean Rooms Differential Privacy only supports queries with Amazon S3-backed AWS Glue tables. It doesn't support queries with Snowflake or Amazon Athena tables.
2. AWS Clean Rooms Differential Privacy doesn't address timing attacks. For example, these attacks are possible in scenarios where an individual user contributes a large number of rows and adding or removing this user significantly changes the query computation time.
3. AWS Clean Rooms Differential Privacy doesn't guarantee differential privacy when a SQL query can result in overflow or invalid cast errors at run time due to the use of certain SQL constructs.

   The following table is a list of some, but not all, SQL constructs that may produce run-time errors and should be verified in analysis templates. We recommend that you approve analysis templates that minimize the chances of such run-time errors and periodically review query logs to determine if the queries align with the collaboration agreement.

   The following SQL constructs are vulnerable to overflow errors:

Category	SQL constructs vulnerable to overflow errors in the Spark SQL analytics engine
Aggregate functions	• AVG

Category	SQL constructs vulnerable to overflow errors in the Spark SQL analytics engine
	• SUM/SUM_DISTINCT
Data type formatting functions	• TO_TIMESTAMP • TO_DATE
Date and time functions	• ADD_MONTHS • DATEADD • DATEDIFF
Math functions	• +, -, *, / • POWER
String functions	• || • CONCAT • REPEAT
Window functions	• AVG • SUM

4. The CAST data type formatting function is vulnerable to invalid cast errors.

   You can configure CloudWatch to create a metric filter for a log group and then create a CloudWatch alarm on that metric filter to receive alerts if a potential overflow or cast error was encountered.

   Specifically, you should monitor for the error codes `CastError`, `OverflowError`, `ConversionError`. The presence of these error codes indicates a potential side-channel attack, but might indicate an erroneous SQL query.

   For more information, see Analysis logging in AWS Clean Rooms.

AWS Clean Rooms ML

AWS Clean Rooms ML allows two or more parties to run machine learning models on their data without the need to share their data with each other. The service provides privacy-enhancing

controls that allow data owners to safe-guard their data and their model IP. You can use AWS authored models or bring your own custom model.

For a more detailed explanation of how this works, see Cross-account jobs.

For more information about the capabilities of Clean Rooms ML models, see the following topics.

Topics

- AWS Clean Rooms ML terminology
- How AWS Clean Rooms ML works with AWS models
- How AWS Clean Rooms ML works with custom models
- AWS models in Clean Rooms ML
- Custom models in Clean Rooms ML

AWS Clean Rooms ML terminology

It is important to understand the following terminology when using Clean Rooms ML:

- *Training data provider* – The party that contributes the training data, creates and configures a lookalike model, and then associates that lookalike model with a collaboration.
- *Seed data provider* – The party that contributes the seed data, generates a lookalike segment, and exports their lookalike segment.
- *Training data* – The training data provider's data, which is used to generate a lookalike model. The training data is used to measure similarity in user behaviors.

  The training data must contain a user ID, item ID, and timestamp column. Optionally, the training data can contain other interactions as numerical or categorical features. Examples of interactions are a list of videos watched, items purchased, or articles read.
- *Seed data* – The seed data provider's data, which is used to create a lookalike segment. The seed data can be provided directly or it can come from the results of an AWS Clean Rooms query. The lookalike segment output is a set of users from the training data that most closely resembles the seed users.
- *Lookalike model* – A machine learning model of the training data that is used to find similar users in other datasets.

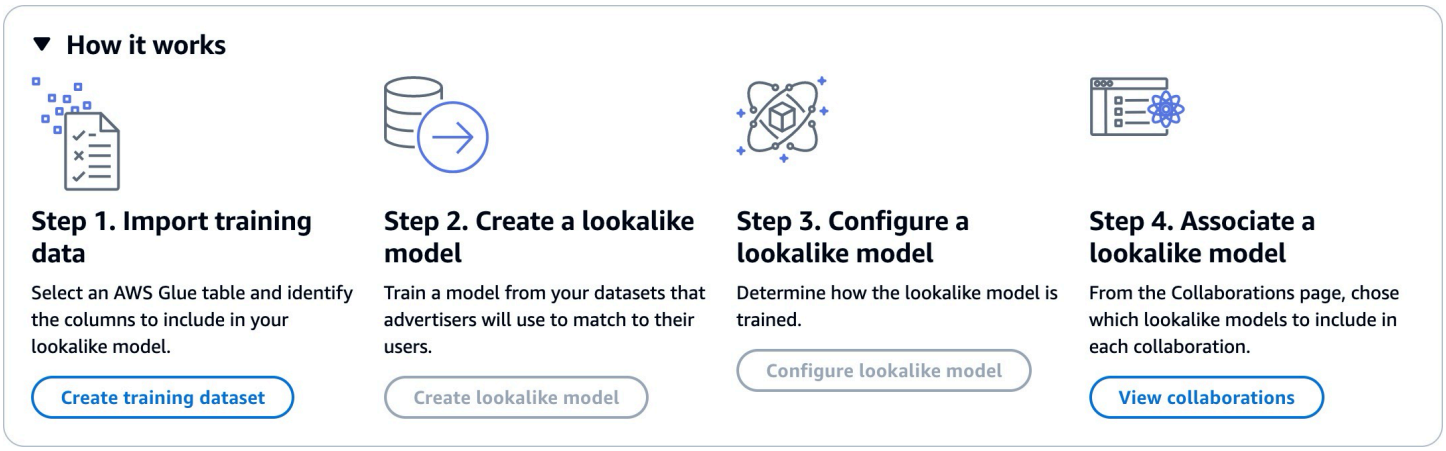
  When using the API, the term *audience model* is used equivalently to lookalike model. For example, you use the CreateAudienceModel API to create a lookalike model.

- *Lookalike segment* – A subset of the training data that most closely resembles the seed data.

  When using the API, you create a lookalike segment with the StartAudienceGenerationJob API.

The training data provider's data is never shared with the seed data provider and the seed data provider's data is never shared with the training data provider. The lookalike segment output is shared with the training data provider, but never the seed data provider.

How AWS Clean Rooms ML works with AWS models



Working with lookalike models requires that two parties, a training data provider and a seed data provider, work sequentially in AWS Clean Rooms to bring their data into a collaboration. This is the workflow that the training data provider must complete first:

1. The training data provider's data must be stored in a AWS Glue data catalog table of user-item interactions. At a minimum, the training data must contain a user ID column, interaction ID column, and a timestamp column.
2. The training data provider registers the training data with AWS Clean Rooms.
3. The training data provider creates a lookalike model that can be shared with multiple seed data providers. The lookalike model is a deep neural network that can take up to 24 hours to train. It isn't automatically retrained and we recommend that you retrain the model weekly.
4. The training data provider configures the lookalike model, including whether to share relevance metrics and the Amazon S3 location of the output segments. The training data provider can create multiple configured lookalike models from a single lookalike model.
5. The training data provider associates the configured audience model to a collaboration that's shared with a seed data provider.

This is the workflow that the seed data provider must complete next:

1. The seed data provider's data can be stored in an Amazon S3 bucket or it can come from the results of query.
2. The seed data provider opens the collaboration that they share with the training data provider.
3. The seed data provider creates a lookalike segment from the Clean Rooms ML tab of the collaboration page.
4. The seed data provider can evaluate the relevance metrics, if they were shared, and export the lookalike segment for use outside AWS Clean Rooms.

How AWS Clean Rooms ML works with custom models

With Clean Rooms ML, members of a collaboration can use a dockerized custom model algorithm that is stored in Amazon ECR to jointly analyze their data. To do this, the *model provider* must create an image and store it in Amazon ECR. Follow the steps in Amazon Elastic Container Registry User Guide to create a private repository that will contain the custom ML model.

Any member of a collaboration can be the *model provider*, provided they have the correct permissions. All members of a collaboration can contribute training data, inference data, or both to the model. For the purpose of this guide, members contributing data are referred to as *data providers*. The member who creates the collaboration is the *collaboration creator*, and this member can be either the *model provider*, one of the *data providers*, or both.

At the highest level, here are the steps that must be completed to perform custom ML modeling:

1. The collaboration creator creates a collaboration and assigns each member the proper member abilities and payment configuration. The collaboration creator must assign the member ability to either receive model outputs or receive inference results to the appropriate member in this step because it can't be updated after the collaboration is created. For more information, see Creating and joining the collaboration in AWS Clean Rooms ML.
2. The model provider configures and associates their containerized ML model to the collaboration and ensures privacy constraints are set for exported data. For more information, see Configuring a model algorithm in AWS Clean Rooms ML.
3. The data providers contribute their data to the collaboration and ensure their privacy needs are specified. Data providers must allow the model to access their data. For more information, see Contributing training data in AWS Clean Rooms ML and Associating the configured model algorithm in AWS Clean Rooms ML.

4. A collaboration member creates the ML configuration, which defines where the model artifacts or inference results are exported to.
5. A collaboration member creates an ML input channel that provides input to the training container or inference container. The ML input channel is a query that defines the data to be used in the context of the model algorithm.
6. A collaboration member invokes model training using the ML input channel and the configured model algorithm. For more information, see Creating a trained model in AWS Clean Rooms ML.
7. (Optional) The model trainer invokes the model export job and the model artifacts are sent to the model results receiver. Only members with a valid ML configuration and the member ability to receive model output can receive model artifacts. For more information, see Exporting model artifacts from AWS Clean Rooms ML.
8. (Optional) A collaboration member invokes model inference using the ML input channel, the trained model ARN, and the inference configured model algorithm. The inference results are sent to the inference output receiver. Only members with a valid ML configuration and the member ability to receive inference output can receive inference results.

Here are the steps that must be completed by the *model provider*:

1. Create a SageMaker AI compatible Amazon ECR docker image. Clean Rooms ML supports only SageMaker AI compatible docker images.
2. After you have created a SageMaker AI compatible docker image, push the image to Amazon ECR. Follow the directions in Amazon Elastic Container Registry User Guide to create a container training image.
3. Configure the model algorithm for use in Clean Rooms ML.
   a. Provide the Amazon ECR repository link and any arguments necessary to configure the model algorithm.
   b. Provide a service access role that allows Clean Rooms ML to access the Amazon ECR repository.
   c. Associate the configured model algorithm with the collaboration. This includes providing a privacy policy that defines controls for container logs, failure logs, CloudWatch metrics, and limits about how much data can be exported from the container results.

Here are the steps that must be completed by the *data provider* to collaborate with a custom ML model:

1. Configure an existing AWS Glue table with a custom analysis rule. This allows a specific set of pre-approved queries or pre-approved accounts to use your data.
2. Associate your configured table with a collaboration and provide a service access role that can access your AWS Glue tables.
3. Add a collaboration analysis rule to the table that allows the configured model algorithm association to access the configured table.
4. After the model and data are associated and configured in Clean Rooms ML, the member with the ability to run queries provides an SQL query and selects the model algorithm to use.

After model training is finished, that member initiates the export of model training artifacts or inference results. These artifacts or results are sent to the member with the ability to received trained model output. The results receiver must configure their `MachineLearningConfiguration` before they can receive model output.

AWS models in Clean Rooms ML

AWS Clean Rooms ML provides a privacy-preserving method for two parties to identify similar users in their data without the need to share their data with each other. The first party brings the training data to AWS Clean Rooms so that they can create and configure a lookalike model and associate it with a collaboration. Then, seed data is brought to the collaboration to create a lookalike segment that resembles the training data.

For a more detailed explanation of how this works, see Cross-account jobs.

The following topics provide information on how to create and configure a AWS models in Clean Rooms ML.

Topics

- Privacy protections of AWS Clean Rooms ML
- Training data requirements for Clean Rooms ML
- Seed data requirements for Clean Rooms ML
- AWS Clean Rooms ML model evaluation metrics

Privacy protections of AWS Clean Rooms ML

Clean Rooms ML is designed to reduce the risk of *membership inference attacks* where the training data provider can learn who is in the seed data and the seed data provider can learn who is in the training data. Several steps are taken to prevent this attack.

First, seed data providers don't directly observe the Clean Rooms ML output and training data providers can never observe the seed data. Seed data providers can choose to include the seed data in the output segment.

Next, the lookalike model is created from a random sample of the training data. This sample includes a significant number of users that don't match the seed audience. This process makes it harder to determine whether a user was not in the data, which is another avenue for membership inference.

Further, multiple seed customers can be used for every parameter of seed-specific lookalike model training. This limits how much the model can overfit, and thus how much can be inferred about a user. As a result, we recommend that the minimum size of the seed data is 500 users.

Finally, user-level metrics are never provided to training data providers, which eliminates another avenue for a membership inference attack.

Training data requirements for Clean Rooms ML

To successfully create a lookalike model, your training data must meet the following requirements:

- The training data must be in Parquet, CSV, or JSON format.

  > **Note**
  >
  > Zstandard (ZSTD) compressed Parquet data is not supported.

- Your training data must be cataloged in AWS Glue. For more information, see Getting started with the AWS Glue Data Catalog in the AWS Glue Developer Guide. We recommend using AWS Glue crawlers to create your tables because the schema is inferred automatically.
- The Amazon S3 bucket that contains the training data and seed data is in the same AWS region as your other Clean Rooms ML resources.
- The training data must contain at least 100,000 unique user IDs with at least two item interactions each.

- The training data must contain at least 1 million records.
- The schema specified in the CreateTrainingDataset action must align with the schema defined when the AWS Glue table was created.
- The required fields, as defined in the provided table, are defined in the CreateTrainingDataset action.

Field type	Supported data types	Required	Description
USER_ID	string, int, bigint	Yes	A unique identifier for each user in the dataset. It should be a non-Personally Identifiable Information (PII) value. This might be a hashed identifier or a customer ID.
ITEM_ID	string, int, bigint	Yes	A unique identifier for each item a user interacts with.
TIMESTAMP	bigint, int, timestamp	Yes	The time when a user

Field type	Supported data types	Required	Descripti on
			interacte d with the item. Values must be in the Unix epoch time in seconds format.

Field type	Supported data types	Required	Description
CATEGORICAL_FEATURE	string, int, float, bigint, double, boolean, array	No	Captures categorical data related to the user or the item. This can include things like an event type (such as click or purchase), user demographics (age group, gender - anonymized), user location (city, country - anonymized), item category (such as clothing or electronics), or item brand.

Field type	Supported data types	Required	Description
NUMERICAL_FEATURE	double, float, int, bigint	No	Captures numerical data related to the user or the item. This can include things like user purchase history (total amount spent), item price, number of times an item is visited, or user ratings for items.

- Optionally, you can provide up to 10 total categorical or numerical features.

Here is an example of a valid training data set in CSV format

```
USER_ID,ITEM_ID,TIMESTAMP,EVENT_TYPE(CATEGORICAL FEATURE),EVENT_VALUE (NUMERICAL
 FEATURE)
196,242,881250949,click,15
186,302,891717742,click,13
22,377,878887116,click,10
244,51,880606923,click,20
166,346,886397596,click,10
```

Seed data requirements for Clean Rooms ML

The seed data for a lookalike model can either come directly from an Amazon S3 bucket or from the results of an SQL query.

Seed data that's provided directly must meet the following requirements:

- The seed data must be in JSON lines format with a list of user IDs.
- The seed size should be between 25 and 500,000 unique user IDs.
- The minimum number of seed users must match the minimum matching seed size value that was specified when you created the configured audience model.

The following is an example of a valid training data set in CSV format

```
{"user_id": "abc"}
{"user_id": "def"}
{"user_id": "ghijkl"}
{"user_id": "123"}
{"user_id": "456"}
{"user_id": "7890"}
```

AWS Clean Rooms ML model evaluation metrics

Clean Rooms ML computes the *recall* and *relevance score* to determine how well your model performs. Recall compares the similarity between the lookalike data and training data. The relevance score is used to decide how large the audience should be, not whether the model is well-performing.

Recall is an unbiased measure of how similar the lookalike segment is to the training data. Recall is the percentage of the most similar users (by default, the most similar 20%) from a sample of the training data that are included in the seed audience by the audience generation job. Values range from 0-1, larger values indicate a better audience. A recall value approximately equal to the maximum bin percentage indicates that the audience model is equivalent to random selection.

We consider this a better evaluation metric than accuracy, precision, and F1 scores because Clean Rooms ML doesn't have accurately labeled true negative users when building its model.

Segment-level *relevance score* is a measure of similarity with values ranging from -1 (least similar) to 1 (most similar). Clean Rooms ML computes a set of relevance scores for various segment

sizes to help you determine the best segment size for your data. Relevance scores monotonically decrease as the segment size increases, thus as the segment size increases it can be less similar to the seed data. When the segment-level relevance score reaches 0, the model predicts that all users in the lookalike segment are from the same distribution as the seed data. Increasing the output size is likely to include users in the lookalike segment that aren't from the same distribution as the seed data.

Relevance scores are normalized within a single campaign and should not be used to compare across campaigns. Relevancy scores shouldn't be used as a single-sourced evidence for any business outcome because those are impacted by multiple complex factors in addition to relevance, such as inventory quality, inventory type, timing of advertising, and so on.

Relevance scores should not be used to judge the quality of the seed, but rather if it can be increased or decreased. Consider the following examples:

- All positive scores – This indicates that there are more output users that are predicted as similar than are included in the lookalike segment. This is common for seed data that's part of a large market, such as everybody who has bought toothpaste in the past month. We recommend looking at smaller seed data, such as everybody who has bought toothpaste more than once in the past month.
- All negatives scores or negative for your desired lookalike segment size – This indicates that Clean Rooms ML predicts there aren't enough similar users in the desired lookalike segment size. This can be because the seed data is too specific or the market is too small. We recommend either applying fewer filters to the seed data or widening the market. For example, if the original seed data was customers that bought a stroller and car seat, you could expand the market to customers that bought multiple baby products.

Training data providers determine whether the relevance scores are exposed and the bucket bins where relevance scores are computed.

Custom models in Clean Rooms ML

With Clean Rooms ML, members of a collaboration can use a dockerized custom model algorithm that is stored in Amazon ECR to jointly analyze their data. To do this, the *model provider* must create an image and store it in Amazon ECR. Follow the steps in Amazon Elastic Container Registry User Guide to create a private repository that will contain the custom ML model.

Any member of a collaboration can be the *model provider*, provided they have the correct permissions. All members of a collaboration can contribute data to the model. For the purpose of this guide, members contributing data are referred to as *data providers*. The member who creates the collaboration is the *collaboration creator*, and this member can be either the *model provider*, one of the *data providers*, or both.

The following topics describe the information necessary to create a custom ML model

Topics

- Custom ML modeling prerequisites
- Model authoring guidelines for the training container
- Model authoring guidelines for the inference container
- Receiving model logs and metrics

Custom ML modeling prerequisites

Before you can perform custom ML modeling, you should consider the following:

- Determine whether both model training and inference on the trained model is going to be performed in the collaboration.
- Determine the role that each collaboration member will perform and assign them the appropriate abilities.
  - Assign the `CAN_QUERY` ability to the member who will train the model and run inference on the trained model.
  - Assign the `CAN_RECEIVE_RESULTS` to at least one member of the collaboration.
  - Assign `CAN_RECEIVE_MODEL_OUTPUT` or `CAN_RECEIVE_INFERENCE_OUTPUT` abilities to the member that will receive trained model exports or inference output, respectively. You can choose to use both abilities if they are required by your use-case.
- Determine the maximum size of the trained model artifacts or inference results that you will allow to be exported.
- We recommend that all users have the `CleanrooomsFullAccess` and `CleanroomsMLFullAccess` policies attached to their role. Using custom ML models requires using both the AWS Clean Rooms and AWS Clean Rooms ML SDKs.
- Consider the following information about IAM roles.

- All data providers must have a service access role that allows AWS Clean Rooms to read data from their AWS Glue catalogs and tables, and the underlying Amazon S3 locations. These roles are similar to those required for SQL querying. This allows you to use the `CreateConfiguredTableAssociation` action. For more information, see Create a service role to create a configured table association.
- All members that want to receive metrics must have a service access role that allows them to write CloudWatch metrics and logs. This role is used by Clean Rooms ML to write all model metrics and logs to the member's AWS account during model training and inference. We also provide privacy controls to determine which members have access to the metrics and logs. This allows you to use the `CreateMLConfiguration` action. For more information see, Create a service role for custom ML modeling - ML Configuration.

  The member receiving results must provide a service access role with permissions to write to their Amazon S3 bucket. This role allows Clean Rooms ML to export results (trained model artifacts or inference results) to an Amazon S3 bucket. This allows you to use the `CreateMLConfiguration` action. For more information, see Create a service role for custom ML modeling - ML Configuration.
- The model provider must provide a service access role with permissions to read their Amazon ECR repository and image. This allows you to use the `CreateConfigureModelAlgorithm` action. For more information, see Create a service role to provide a custom ML model.
- The member that creates the `MLInputChannel` to generate datasets for training or inference must provide a service access role that allows Clean Rooms ML to execute an SQL query in AWS Clean Rooms. This allows you to use the `CreateTrainedModel` and `StartTrainedModelInferenceJob` actions. For more information, see Create a service role to query a dataset.

- Model authors should follow the Model authoring guidelines for the training container and Model authoring guidelines for the inference container to ensure model inputs and outputs are configured as expected by AWS Clean Rooms.

Model authoring guidelines for the training container

This section details the guidelines that model providers should follow when creating a custom ML model algorithm for Clean Rooms ML.

- Use the appropriate SageMaker AI training-supported container base image, as described in the SageMaker AI Developer Guide. The following code allows you to pull the supported container base images from public SageMaker AI endpoints.

```
ecr_registry_endpoint='763104351884.dkr.ecr.$REGION.amazonaws.com'
base_image='pytorch-training:2.3.0-cpu-py311-ubuntu20.04-sagemaker'
aws ecr get-login-password --region $REGION | docker login --username AWS --password-
stdin $ecr_registry_endpoint
docker pull $ecr_registry_endpoint/$base_image
```

- When authoring the model locally, ensure the following so that you can test your model locally, on a development instance, on SageMaker AI Training in your AWS account, and on Clean Rooms ML.
  - We recommend writing a training script that accesses useful properties about the training environment through various environment variables. Clean Rooms ML uses the following arguments to invoke training on your model code: `SM_MODEL_DIR`, `SM_OUTPUT_DIR`, `SM_CHANNEL_TRAIN`, and `FILE_FORMAT`. These defaults are used by Clean Rooms ML to train your ML model in its own execution environment with the data from all parties.
  - Clean Rooms ML makes your training input channels available via the `/opt/ml/input/data/`*`channel-name`* directories in the docker container. Each ML input channel is mapped based on its corresponding `channel_name` provided in the `CreateTrainedModel` request.

```
parser = argparse.ArgumentParser()# Data, model, and output directories

parser.add_argument('--model_dir', type=str, default=os.environ.get('SM_MODEL_DIR',
 "/opt/ml/model"))
parser.add_argument('--output_dir', type=str,
 default=os.environ.get('SM_OUTPUT_DIR', "/opt/ml/output/data"))
parser.add_argument('--train_dir', type=str,
 default=os.environ.get('SM_CHANNEL_TRAIN', "/opt/ml/input/data/train"))
parser.add_argument('--train_file_format', type=str,
 default=os.environ.get('FILE_FORMAT', "csv"))
```

  - Ensure that you are able to generate a synthetic or test dataset based on the schema of the collaborators that will be used in your model code.
  - Ensure that you can run a SageMaker AI training job on your own AWS account before you associate the model algorithm with a AWS Clean Rooms collaboration.

The following code contains a sample Docker file that is compatible with local testing, SageMaker AI Training environment testing, and Clean Rooms ML

```
FROM  763104351884.dkr.ecr.us-west-2.amazonaws.com/pytorch-training:2.3.0-cpu-
py311-ubuntu20.04-sagemaker
MAINTAINER $author_name

ENV PYTHONDONTWRITEBYTECODE=1 \
    PYTHONUNBUFFERED=1 \
    LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:/usr/local/lib"

ENV PATH="/opt/ml/code:${PATH}"

# this environment variable is used by the SageMaker PyTorch container to determine
 our user code directory
ENV SAGEMAKER_SUBMIT_DIRECTORY /opt/ml/code

# copy the training script inside the container
COPY train.py /opt/ml/code/train.py
# define train.py as the script entry point
ENV SAGEMAKER_PROGRAM train.py
ENTRYPOINT ["python", "/opt/ml/code/train.py"]
```

- To best monitor container failures, we recommend exporting logs and debugging for failure reasons. In a `GetTrainedModel` response, Clean Rooms ML returns the first 1024 characters from this file under `StatusDetails`.
- After you have completed any model changes and you are ready to test it in the SageMaker AI environment, run the following commands in the order provided.

```
export ACCOUNT_ID=xxx
export REPO_NAME=xxx
export REPO_TAG=xxx
export REGION=xxx

docker build -t $ACCOUNT_ID.dkr.ecr.us-west-2.amazonaws.com/$REPO_NAME:$REPO_TAG

# Sign into AWS $ACCOUNT_ID/ Run aws configure
# Check the account and make sure it is the correct role/credentials
aws sts get-caller-identity
aws ecr create-repository --repository-name $REPO_NAME --region $REGION
aws ecr describe-repositories --repository-name $REPO_NAME --region $REGION
```

```
# Authenticate Doker
aws ecr get-login-password --region $REGION | docker login --username AWS --password-
stdin $ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com

# Push To ECR Images
docker push  $ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com$REPO_NAME:$REPO_TAG

# Create Sagemaker Training job
# Configure the training_job.json with
# 1. TrainingImage
# 2. Input DataConfig
# 3. Output DataConfig
aws sagemaker create-training-job --cli-input-json file://training_job.json --region
 $REGION
```

After the SageMaker AI job is complete and you are satisfied with your model algorithm, you can register the Amazon ECR Registry with AWS Clean Rooms ML. Use the `CreateConfiguredModelAlgorithm` action to register the model algorithm and the `CreateConfiguredModelAlgorithmAssociation` to associate it to a collaboration.

Model authoring guidelines for the inference container

This section details the guidelines that model providers should follow when creating an inference algorithm for Clean Rooms ML.

- Use the appropriate SageMaker AI inference-supported container base image, as described in the SageMaker AI Developer Guide. The following code allows you to pull the supported container base images from public SageMaker AI endpoints.

```
ecr_registry_endpoint='763104351884.dkr.ecr.$REGION.amazonaws.com'
base_image='pytorch-inference:2.3.0-cpu-py311-ubuntu20.04-sagemaker'
aws ecr get-login-password --region $REGION | docker login --username AWS --password-
stdin $ecr_registry_endpoint
docker pull $ecr_registry_endpoint/$base_image
```

- When authoring the model locally, ensure the following so that you can test your model locally, on a development instance, on SageMaker AI Batch Transform in your AWS account, and on Clean Rooms ML.

- Clean Rooms ML makes your model artifacts from inference available for use by your inference code via the `/opt/ml/model` directory in the docker container.
- Clean Rooms ML splits input by line, uses a `MultiRecord` batch strategy, and adds a newline character at the end of every transformed record.
- You set the maximum request size in `maxPayloadInMB` under `containerExecutionParameters`. If you don't specify a value, Clean Rooms ML uses 6 MB. When an input CSV file is larger than that value, SageMaker AI splits the file into multiple requests to your inference container. The CSV header row appears only in the first request for that file, and subsequent requests begin with data rows. Configure your inference container with a predefined column schema so that it accepts requests with or without a header row. If a header row is present, ignore it instead of treating it as a data record. For more information about `maxPayloadInMB`, see InferenceContainerExecutionParameters in the *AWS Clean Rooms ML API Reference*.
- Ensure that you are able to generate a synthetic or test inference dataset based on the schema of the collaborators that will be used in your model code.
- Ensure that you can run a SageMaker AI batch transform job on your own AWS account before you associate the model algorithm with a AWS Clean Rooms collaboration.

  The following code contains a sample Docker file that is compatible with local testing, SageMaker AI transform environment testing, and Clean Rooms ML

```
FROM 763104351884.dkr.ecr.us-east-1.amazonaws.com/pytorch-inference:1.12.1-cpu-
py38-ubuntu20.04-sagemaker

ENV PYTHONUNBUFFERED=1

COPY serve.py /opt/ml/code/serve.py
COPY inference_handler.py /opt/ml/code/inference_handler.py
COPY handler_service.py /opt/ml/code/handler_service.py
COPY model.py /opt/ml/code/model.py

RUN chmod +x /opt/ml/code/serve.py

ENTRYPOINT ["/opt/ml/code/serve.py"]
```

- After you have completed any model changes and you are ready to test it in the SageMaker AI environment, run the following commands in the order provided.

```
export ACCOUNT_ID=xxx
export REPO_NAME=xxx
export REPO_TAG=xxx
export REGION=xxx

docker build -t $ACCOUNT_ID.dkr.ecr.us-west-2.amazonaws.com/$REPO_NAME:$REPO_TAG

# Sign into AWS $ACCOUNT_ID/ Run aws configure
# Check the account and make sure it is the correct role/credentials
aws sts get-caller-identity
aws ecr create-repository --repository-name $REPO_NAME --region $REGION
aws ecr describe-repositories --repository-name $REPO_NAME --region $REGION

# Authenticate Docker
aws ecr get-login-password --region $REGION | docker login --username AWS --password-
stdin $ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com

# Push To ECR Repository
docker push $ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com$REPO_NAME:$REPO_TAG

# Create Sagemaker Model
# Configure the create_model.json with
# 1. Primary container -
    # a. ModelDataUrl - S3 Uri of the model.tar from your training job
aws sagemaker create-model --cli-input-json file://create_model.json --region $REGION

# Create Sagemaker Transform Job
# Configure the transform_job.json with
# 1. Model created in the step above
# 2. MultiRecord batch strategy
# 3. Line SplitType for TransformInput
# 4. AssembleWith Line for TransformOutput
aws sagemaker create-transform-job --cli-input-json file://transform_job.json --
region $REGION
```

After the SageMaker AI job is complete and you are satisfied with your batch transform, you can register the Amazon ECR Registry with AWS Clean Rooms ML. Use the `CreateConfiguredModelAlgorithm` action to register the model algorithm and the `CreateConfiguredModelAlgorithmAssociation` to associate it to a collaboration.

Receiving model logs and metrics

To receive logs and metrics from custom model training or inference, members must have created an ML Configuration with a valid role that provides the necessary CloudWatch permissions (see Create a service role for custom ML modeling - ML Configuration).

System metric

System metrics for both training and inference, such as CPU and memory utilization, are published to all members in the collaboration with valid ML Configurations. These metrics can be viewed as the job progresses via CloudWatch Metrics in the `/aws/cleanroomsml/TrainedModels` or `/aws/cleanroomsml/TrainedModelInferenceJobs` namespaces, respectively.

Model logs

Access to the model logs is provided by the privacy configuration policy of each configured model algorithm. The model author sets the privacy configuration policy when associating a configured model algorithm (either via the console or the `CreateConfiguredModelAlgorithmAssociation` API) to a collaboration. Setting the privacy configuration policy controls which members can receive the model logs.

Additionally, the model author can set a filter pattern in the privacy configuration policy to filter log events. All logs that a model container sends to `stdout` or `stderr` and that match the filter pattern (if set), are sent to Amazon CloudWatch Logs. Model logs are available in CloudWatch log groups `/aws/cleanroomsml/TrainedModels` or `/aws/cleanroomsml/TrainedModelInferenceJobs`, respectively.

Custom defined metrics

When you configure a model algorithm (either via the console or the `CreateConfiguredModelAlgorithm` API), the model author can provide specific metric names and regex statements to search for in the output logs. These can be viewed as the job progresses via CloudWatch Metrics in the `/aws/cleanroomsml/TrainedModels` namespace. When associating a configured model algorithm, the model author can set an optional noise level in the metrics privacy configuration to avoid outputting raw data while still providing visibility into custom metric trends. If a noise level is set, the metrics are published at the end of the job rather than in real time.

Cryptographic Computing for Clean Rooms

Cryptographic Computing for Clean Rooms (C3R) is a capability in AWS Clean Rooms that can be used in addition to analysis rules. With C3R, organizations can bring sensitive data together to derive new insights from data analytics while cryptographically limiting what can be learned by any party in the process. C3R can be used by two or more parties that want to collaborate with their sensitive data but are required to only use encrypted data in the cloud.

The C3R encryption client is a client-side encryption tool that you can use to encrypt your data for use with AWS Clean Rooms. When you use the C3R encryption client, data remains cryptographically protected while in use in an AWS Clean Rooms collaboration. As with a regular AWS Clean Rooms collaboration, the input data is relational database tables, and the computation is expressed as a SQL query. However, C3R only supports a limited subset of SQL queries on encrypted data.

Specifically, C3R supports SQL JOIN and SELECT statements on cryptographically protected data. Each column in the input table can be used in exactly one of the following SQL statement types:

- Columns that are cryptographically protected for use in JOIN statements are called **fingerprint columns**.
- Columns that are cryptographically protected for use in SELECT statements are called **sealed columns**.
- Columns that are not cryptographically protected for use in JOIN or SELECT statements are called **cleartext columns**.

In some cases, GROUP BY statements are supported on fingerprint columns. For more information, see Fingerprint columns. Currently, C3R doesn't support the use of other SQL constructs on encrypted data, such as WHERE clauses or aggregate functions like SUM and AVERAGE, even if they would otherwise be allowed by the relevant analysis rules.

C3R is designed to protect data in individual cells of a table. Using the default configuration for C3R, the underlying data that a customer makes available to third parties through a collaboration remains encrypted while the content is in use within AWS Clean Rooms. C3R uses industry standard AES-GCM encryption for all sealed columns and an industry standard pseudorandom function, known as a Hash-based Message Authentication Code (HMAC), for protecting fingerprint columns.

Although C3R encrypts the data in your tables, the following information might still be able to be inferred:

- Information about the tables themselves, including the number of columns, column names, and the number of rows in your table.
- As with most standard forms of encryption, C3R doesn't try to hide the length of encrypted values. C3R does offer the ability to pad encrypted values to hide the exact length of cleartexts. However, an upper bound on the length of the cleartexts in each column could still be revealed to another party.
- Logging-level information, such as when a particular row was added to an encrypted C3R table.

For more information about C3R, see the following topics.

Topics

- Considerations when using Cryptographic Computing for Clean Rooms
- Supported file and data types in Cryptographic Computing for Clean Rooms
- Column names in Cryptographic Computing for Clean Rooms
- Column types in Cryptographic Computing for Clean Rooms
- Cryptographic computing parameters
- Optional flags in Cryptographic Computing for Clean Rooms
- Queries with Cryptographic Computing for Clean Rooms
- Guidelines for the C3R encryption client

Considerations when using Cryptographic Computing for Clean Rooms

Cryptographic Computing for Clean Rooms (C3R) seeks to maximize data protection. However, some use cases might benefit from lower levels of data protection in exchange for additional functionality. You can make these specific tradeoffs by modifying C3R from its most secure configuration. As the customer, you should be aware of these tradeoffs and determine if they are appropriate for your use case. Tradeoffs to consider include the following:

Topics

- Allowing mixed cleartext and encrypted data in your tables
- Allowing repeated values in fingerprint columns
- Loosening restrictions on how fingerprint columns are named
- Determining how NULL values are represented

For more information about how to set parameters for these scenarios, see Cryptographic computing parameters.

Allowing mixed cleartext and encrypted data in your tables

Having all data be client-side encrypted provides maximum data protection. However, this limits certain kinds of queries (for example, the SUM aggregate function). The risk of allowing cleartext data is that it's feasible that anyone with access to the encrypted tables could infer some information about encrypted values. This could be done by performing a statistical analysis on the cleartext and associated data.

For example, imagine you had the columns of `City` and `State`. The `City` column is cleartext and the `State` column is encrypted. When you see the value `Chicago` in the `City` column, that helps you determine with high probability that the `State` is `Illinois`. In contrast, if one column is `City` and the other column is `EmailAddress`, a cleartext `City` is unlikely to reveal anything about an encrypted `EmailAddress`.

For more information about the parameter for this scenario, see Allow cleartext columns parameter.

Allowing repeated values in fingerprint columns

For the most secure approach, we assume that any fingerprint column contains exactly one instance of a variable. No item can be repeated in a fingerprint column. The C3R encryption client maps these cleartext values into unique values that are indistinguishable from random values. Therefore, it's impossible to infer information about the cleartext from these random values.

The risk of repeated values in a fingerprint column is that repeated values will result in repeated random-looking values. Thus, anyone who has access to the encrypted tables could, in theory, perform a statistical analysis of the fingerprint columns that might reveal information about cleartext values.

Again, suppose the fingerprint column is `State`, and every row of the table corresponds to a US household. By doing a frequency analysis, one could infer which state is `California` and which is `Wyoming` with high probability. This inference is possible because `California` has many more residents than `Wyoming`. In contrast, say the fingerprint column is on a household identifier and each household appeared in the database between 1 and 4 times in a database of millions of entries. It's unlikely that a frequency analysis would reveal any useful information.

For more information about the parameter for this scenario, see Allow duplicates parameter.

Loosening restrictions on how fingerprint columns are named

By default, we assume that when two tables are joined using encrypted fingerprint columns, those columns have the same name in each table. The technical reason for this result is that, by default, we derive a different cryptographic key for encrypting each fingerprint column. That key is derived from a combination of the shared secret key for the collaboration and the column name. If we try to join two columns with different column names, we derive different keys and we can't compute a valid join.

To address this issue, you can turn off the feature that derives keys from each column name. Then, the C3R encryption client uses a single derived key for all fingerprint columns. The risk is that another kind of frequency analysis can be done that might reveal information.

Let's use the `City` and `State` example again. If we derive the same random values for each fingerprint column (by not incorporating the column name). `New York` has the same random value in the `City` and `State` columns. New York is one of a few cities in the US where the `City` name is the same as the `State` name. In contrast, if your dataset has completely different values in each column, no information is leaked.

For more information about the parameter for this scenario, see Allow JOIN of columns with different names parameter.

Determining how NULL values are represented

The option available to you is whether to process cryptographically (encrypt and HMAC) NULL values like any other value. If you don't process NULL values like any other value, information might be revealed.

For example, suppose that NULL in the `Middle Name` column in the cleartext indicates people without middle names. If you don't encrypt those values, you leak which rows in the encrypted table are used for people without middle names. That information might be an identifying signal for some people in some populations. But if you do cryptographically process NULL values, certain SQL queries act differently. For example, GROUP BY clauses will not group fingerprint NULL values in fingerprint columns together.

For more information about the parameter for this scenario, see Preserve NULL values parameter.

Supported file and data types in Cryptographic Computing for Clean Rooms

The C3R encryption client recognizes the following file types:

- CSV files
- Parquet files

You can use the `--fileFormat` flag in the C3R encryption client to specify a file format explicitly. When explicitly specified, file format is not determined by file extension.

Topics

- CSV files
- Parquet files
- Encrypting non-string values

CSV files

A file with a .csv extension is assumed to be CSV formatted and contain UTF-8 encoded text. The C3R encryption client treats all values as strings.

Supported properties in .csv files

The C3R encryption client requires that .csv files have the following properties:

- Might or might not contain an initial header row that uniquely names each column.
- Comma-delimited. (Currently, custom delimiters are not supported.)
- UTF-8 encoded text.

White space trimming from .csv entries

Both leading and trailing white space is trimmed from .csv entries.

Custom NULL encoding for a .csv file

A .csv file can use custom NULL encoding.

With the C3R encryption client, you can specify custom encodings for NULL entries in the input data by using the `--csvInputNULLValue=<csv-input-null>` flag. The C3R encryption client can use custom encodings in the generated output file for NULL entries by using the `--csvOutputNULLValue=<csv-output-null>` flag.

> **Note**
>
> A NULL entry is considered to be *lacking* content, specifically in the context of a richer tabular format like an SQL table. Although .csv doesn't explicitly support this characterization for historical reasons, it's a common convention to consider an empty entry that contains only white space to be NULL. Therefore, that's the default behavior of the C3R encryption client and it can be customized as needed.

How .csv entries are interpreted by C3R

The following table provides examples of how .csv entries are marshalled (cleartext to cleartext for clarity) based on the values (if any) that are provided for the `--csvInputNULLValue=<csv-input-null>` and `--csvOutputNULLValue=<csv-output-null>` flags. Leading and trailing white space outside of quotes is trimmed before C3R interprets any value's meaning.

<csv-input-null>	**<csv-output-null>**	**Input entry**	**Output entry**
None	None	`,AnyProduct,`	`,AnyProduct,`
None	None	`, AnyProduct ,`	`,AnyProduct,`
None	None	`,"AnyProduct",`	`,AnyProduct,`
None	None	`, "AnyProduct" ,`	`,AnyProduct,`
None	None	`,,`	`,,`
None	None	`, ,`	`,,`
None	None	`,"",`	`,,`
None	None	`," ",`	`," ",`

<csv-input-null>	<csv-output-null>	Input entry	Output entry
None	None	, " " ,	," ",
"AnyProduct"	"NULL"	,AnyProduct,	,NULL,
"AnyProduct"	"NULL"	, AnyProduct ,	,NULL,
"AnyProduct"	"NULL"	,"AnyProduct",	,NULL,
"AnyProduct"	"NULL"	, "AnyProduct" ,	,NULL,
None	"NULL"	,,	,NULL,
None	"NULL"	, ,	,NULL,
None	"NULL"	,"",	,NULL,
None	"NULL"	," ",	," ",
None	"NULL"	, " " ,	," ",
""	"NULL"	,,	,NULL,
""	"NULL"	, ,	,NULL,
""	"NULL"	,"",	,"",
""	"NULL"	," ",	," ",
""	"NULL"	, " " ,	," ",
"\"\""	"NULL"	,,	,,
"\"\""	"NULL"	, ,	,,
"\"\""	"NULL"	,"",	,NULL,
"\"\""	"NULL"	," ",	," ",

<csv-input-null>	<csv-output-null>	Input entry	Output entry
"\"\""	"NULL"	, " " ,	," ",

CSV file without headers

The source .csv file doesn't need to have headers in the first row that uniquely name each column. However, a .csv file without a header row requires a positional encryption schema. The positional encryption schema is required instead of the typical mapped schema that's used for both .csv files with a header row and Parquet files.

A positional encryption schema specifies output columns by position instead of by name. A mapped encryption schema maps source column names to target column names. For more information, including a detailed discussion and examples of both schema formats, see Mapped and positional table schemas.

Parquet files

A file with a .parquet extension is assumed to be in the Apache Parquet format.

Supported Parquet data types

The C3R encryption client can process any non-complex (that is, primitive type) data in a Parquet file that represents a data type supported by AWS Clean Rooms.

However, only string columns can be used for sealed columns.

The following Parquet data types are supported:

- `Binary` primitive type with the following logical annotations:
  - None if the `--parquetBinaryAsString` is set (`STRING` data type)
  - `Decimal(scale, precision)` (`DECIMAL` data type)
  - `String` (`STRING` data type)
- `Boolean` primitive data type with no logical annotation (`BOOLEAN` data type)
- `Double` primitive data type with no logical annotation (`DOUBLE` data type)
- `Fixed_Len_Binary_Array` primitive type with the `Decimal(scale, precision)` logical annotation (`DECIMAL` data type)

- `Float` primitive data type with no logical annotation (`FLOAT` data type)
- `Int32` primitive type with the following logical annotations:
  - None (`INT` data type)
  - `Date` (`DATE` data type)
  - `Decimal(scale, precision)` (`DECIMAL` data type)
  - `Int(16, true)` (`SMALLINT` data type)
  - `Int(32, true)` (`INT` data type)
- `Int64` primitive data type with the following logical annotations:
  - None (`BIGINT` data type)
  - `Decimal(scale, precision)` (`DECIMAL` data type)
  - `Int(64, true)` (`BIGINT` data type)
  - `Timestamp(isUTCAdjusted, TimeUnit.MILLIS)` (`TIMESTAMP` data type)
  - `Timestamp(isUTCAdjusted, TimeUnit.MICROS)` (`TIMESTAMP` data type)
  - `Timestamp(isUTCAdjusted, TimeUnit.NANOS)` (`TIMESTAMP` data type)

Encrypting non-string values

Currently, only string values are supported for sealed columns.

For .csv files, the C3R encryption client treats all values as UTF-8 encoded text and makes no attempt to interpret them differently before encryption.

For fingerprint columns, types are grouped into equivalence classes. An equivalence class is a set of data types that can be unambiguously compared for equality via a representative data type.

Equivalence classes allow identical fingerprints to be assigned to the same semantic value regardless of the original representation. However, the same value in two equivalence classes will not result in the same fingerprint column.

For example, the `INTEGRAL` value `42` will be assigned the same fingerprint regardless of whether it was originally an `SMALLINT`, `INT`, or `BIGINT`. Also, the `INTEGRAL` value `0` will never match the `BOOLEAN` value `FALSE` (which is represented by the value `0`).

The following equivalence classes and corresponding AWS Clean Rooms data types are supported by fingerprint columns:

Equivalence class	Supported AWS Clean Rooms data type
`BOOLEAN`	`BOOLEAN`
`DATE`	`DATE`
`INTEGRAL`	`BIGINT`, `INT`, `SMALLINT`
`STRING`	`CHAR`, `STRING`, `VARCHAR`

Column names in Cryptographic Computing for Clean Rooms

By default, the names of columns are important in Cryptographic Computing for Clean Rooms.

If the value of the **Allow JOIN of columns with different names** parameter is **false**, column names are used during the encryption of fingerprint columns. For this reason, by default, collaborators must coordinate in advance and use the same target column names for data that will use JOIN statements in queries. By default, columns encrypted for JOIN with different names don't successfully JOIN on any values.

If the value of the **Allow JOIN of columns with different names** parameter is **true**, JOIN statements across columns encrypted as fingerprint columns succeed. Encrypting data with this parameter might allow some inference of the cleartext values. For example, if a row has the same Hash-based Message Authentication Code (HMAC) value in both the `City` column and `State` column, the value might be `New York`.

Normalization of column header names

Column header names are normalized by the C3R encryption client. Any leading and trailing white space is removed, and the column name is made lowercase for the transformed output.

Normalization is applied before all other computations, calculations, or other operations which could possibly be impacted by column names. The emitted output file only contains the normalized names.

Column types in Cryptographic Computing for Clean Rooms

This topic provides information about column types in Cryptographic Computing for Clean Rooms.

Topics

- Fingerprint columns
- Sealed columns
- Cleartext columns

Fingerprint columns

Fingerprint columns are columns that are protected cryptographically for use in JOIN statements.

Data from fingerprint columns can't be decrypted. Only data from sealed columns can be decrypted.

Fingerprint columns must only be used in the following SQL clauses and functions:

- JOIN (INNER, OUTER, LEFT, RIGHT, or FULL) against other fingerprint columns:
  - If the value of the `allowJoinsOnColumnsWithDifferentNames` parameter is set to `false`, both fingerprint columns of the JOIN must also have the same name.
- `SELECT COUNT()`
- `SELECT COUNT(DISTINCT )`
- `GROUP BY` (Only use if the collaboration has set the value of the `preserveNulls` parameter to `true`.)

Queries that violate these constraints might yield incorrect results.

Sealed columns

Sealed columns are columns that are protected cryptographically for use in SELECT statements.

Sealed columns must only be used in the following SQL clauses and functions:

- `SELECT`
- `SELECT ... AS`
- `SELECT COUNT()`

> **Note**
>
> `SELECT COUNT(DISTINCT )` is not supported.

Queries that violate these constraints might yield incorrect results.

Padding data for a sealed column before encryption

When you specify that a column should be a sealed column, C3R asks you what kind of *padding* to choose. Padding data before encryption is optional. Without padding (a pad type of none), the encrypted data's length indicates the size of the cleartext. In some circumstances, the size of the cleartext could expose the plaintext. With padding (a pad type of `fixed` or `max`), all values are first padded to a common size and then encrypted. With padding, the length of the encrypted data provides no information about the original cleartext length, other than giving an upper bound on its size.

If you want padding for a column and the maximal byte length of data in that column is known, use `fixed` padding. Use a `length` value that is at least as large as the byte-length of the longest value in that column.

> **Note**
>
> An error occurs and encryption fails if a value is longer than the provided `length`.

If you want padding for a column and the maximal byte length of data in that column isn't known, use `max` padding. This padding mode pads all data to the length of the longest value plus additional `length` bytes.

> **Note**
>
> You might want to encrypt data in batches, or update your tables with new data periodically. Be aware that `max` padding will pad entries to the length (plus `length` byte) of the longest plaintext entry in a given batch. This means that the ciphertext length may vary from batch to batch. Therefore, if you know the maximum byte-length for a column, then you should use `fixed` instead of `max`.

Cleartext columns

Cleartext columns are columns that aren't protected cryptographically for use in JOIN or SELECT statements.

Cleartext columns can be used in any part of the SQL query.

Cryptographic computing parameters

Cryptographic computing parameters are available for collaborations using Cryptographic Computing for Clean Rooms (C3R) when creating a collaboration. You can create a collaboration using either the AWS Clean Rooms console or the `CreateCollaboration` API operation. In the console, you can set values for the parameters in **Cryptographic computing parameters** after you turn on the **Support cryptographic computing** option. For more information, see the following topics.

Topics

- Allow cleartext columns parameter
- Allow duplicates parameter
- Allow JOIN of columns with different names parameter
- Preserve NULL values parameter

Allow cleartext columns parameter

In the console, you can set the **Allow cleartext columns** parameter when creating a collaboration to specify if cleartext data is allowed in a table with encrypted data.

The following table describes the values for the **Allow cleartext columns** parameter.

Parameter value	Description
No	Cleartext columns aren't allowed in the encrypted table. All data is cryptographically protected.
Yes	Cleartext columns are allowed in the encrypted table. Cleartext columns are not cryptographically protected and are included as cleartext. You should take note of what your rows' cleartext data might reveal about the other data in the table. To run SUM or AVG on specific columns, the columns must be in cleartext.

Using the `CreateCollaboration` API operation, for the `dataEncryptionMetadata` parameter, you can set the value of `allowCleartext` to `true` or `false`. For more information about API operations, see the AWS Clean Rooms API Reference.

Cleartext columns correspond to columns that are classified as cleartext in the table-specific schema. Data in these columns is not encrypted and can be used in any way. Cleartext columns can be useful if the data is not sensitive and/or if more flexibility is needed than an encrypted sealed column or fingerprint column allows.

Allow duplicates parameter

In the console, you can set the **Allow duplicates** parameter when creating a collaboration to specify if columns encrypted for JOIN queries can contain duplicate non-NULL values.

> **Important**
>
> The **Allow duplicates**, **Allow JOIN of columns with different names**, and **Preserve NULL values** parameters have separate but related effects.

The following table describes the values for the **Allow duplicates** parameter.

Parameter value	Description
No	Repeated values are not allowed in a fingerprint column. All values in a single fingerprint column must be unique.
Yes	Repeated values are allowed in a fingerprint column. If you need to join columns with repeated values, set this value to **Yes**. When set to **Yes**, frequency patterns appearing within fingerprint columns in the C3R table or results might imply some additional information about the structure of the cleartext data.

Using the `CreateCollaboration` API operation, for the `dataEncryptionMetadata` parameter you can set the value of `allowDuplicates` to `true` or `false`. For more information about API operations, see the AWS Clean Rooms API Reference.

By default, if encrypted data must be used in JOIN queries, the C3R encryption client requires that those columns have no duplicate values. This requirement is an effort to increase data protection. This behavior can help ensure that repeated patterns in the data are not observable. However, if you want to work with encrypted data in JOIN queries and aren't concerned about duplicate values, the **Allow duplicates** parameter can disable this conservative check.

Allow JOIN of columns with different names parameter

In the console, you can set the **Allow JOIN of columns with different names** parameter when creating a collaboration to specify if JOIN statements between columns with different names are supported.

For more information, see Normalization of column header names

The following table describes the values for the **Allow JOIN of columns with different names** parameter.

Parameter value	Description
No	Joins of fingerprint columns with different names are not supported. JOIN statements only provide accurate results on columns that have the same name. **Important** The **No** value provides increased information security but requires collaboration participants to agree previously about column names. If two columns have different names when encrypted as fingerprint columns and **Allow JOIN of columns with different names** is set to **No**, JOIN statements on those columns produce no results. This is because no values post-encryption are shared between them.
Yes	Joins of fingerprint columns with different names are supported. For additional flexibility, users can set this value to **Yes**, which allows JOIN statements on columns regardless of their column name.

Parameter value	Description
	If set to **Yes**, the C3R encryption client doesn't consider the column name when protecting fingerprint columns. As a result, common values across different fingerprint columns are observable in the C3R table. For example, if a row has the same encrypted JOIN value in both a `City` column and a `State` column, it might be reasonable to infer that value is `New York`.

Using the `CreateCollaboration` API operation, for the `dataEncryptionMetadata` parameter, you can set the value of `allowJoinsOnColumnsWithDifferentNames` to `true` or `false`. For more information about API operations, see the AWS Clean Rooms API Reference.

By default, fingerprint column encryption is affected by the `targetHeader` for that column, set in Step 4: Generate an encryption schema for a tabular file. Therefore, the same cleartext value has different encrypted representations in each different fingerprint column that it's encrypted for.

This parameter can be useful at preventing the inference of cleartext values in some cases. For example, seeing the same encrypted value in fingerprint columns `City` and `State` might be used to reasonably infer the value is `New York`. However, this parameter's use requires additional coordination in advance, so that all columns to be joined in queries have shared names.

You can use the **Allow JOIN of columns with different names** parameter to loosen this restriction. When the parameter value is set to `Yes`, it allows any columns encrypted for JOIN to be used together regardless of name.

Preserve NULL values parameter

In the console, you can set the **Preserve NULL values** parameter when creating a collaboration to indicate that there is no value present for that column.

The following table describes the values for the **Preserve NULL values** parameter.

Parameter value	Description
No	NULL values are not preserved. NULL values don't appear as NULL in an encrypted table. NULL values appear as unique random values in a C3R table.
Yes	NULL values are preserved. NULL values appear as NULL in an encrypted table. If you require SQL semantics of NULL values, you can set this value to **Yes**. As a result, NULL entries appear as NULL in the C3R table, regardless of whether the column is encrypted and regardless of the parameter setting for **Allow duplicates**.

Using the `CreateCollaboration` API operation, for the `dataEncryptionMetadata` parameter, you can set the value of `preserveNulls` to `true` or `false`. For more information about API operations, see the AWS Clean Rooms API Reference.

When the **Preserve NULL values** parameter is set to **No** for the collaboration:

1. NULL entries in `cleartext` columns are unchanged.
2. NULL entries in encrypted `fingerprint` columns are encrypted as random values to conceal their contents. Joining on an encrypted column with NULL entries in the cleartextcolumn doesn't produce any matches for any of the NULL entries. No matches are made because they each receive their own, unique random content.
3. NULL entries in encrypted `sealed` columns are encrypted.

When the value of the **Preserve NULL values** parameter is set to **Yes** for the collaboration, NULL entries from all columns remain as NULL regardless of whether the column is encrypted.

The **Preserve NULL values** parameter is useful in scenarios such as data enrichment, where you want to share a lack of information expressed as NULL. The **Preserve NULL values** parameter is also useful in fingerprint or HMAC format if you have NULL values in the column you want to JOIN or GROUP BY.

If the value of the **Allow duplicates** and **Preserve NULL values** parameters is set to **No**, having more than one NULL entry in a fingerprint column produces an error and stops encryption. If the value of either parameter is set to **Yes**, no such error occurs.

Optional flags in Cryptographic Computing for Clean Rooms

The following sections describe the optional flags that you can set when you encrypt data using the C3R encryption client for tabular file customization and testing.

Topics

- --csvInputNULLValue flag
- --csvOutputNULLValue flag
- --enableStackTraces flag
- --dryRun flag
- --tempDir flag

`--csvInputNULLValue` flag

You can use the `--csvInputNULLValue` flag to specify custom encodings for NULL entries in the input data when you encrypt data using the C3R encryption client.

The following table summarizes the usage and parameters of this flag.

Usage	Parameters
Optional. Users can specify custom encodings for NULL entries in the input data.	User-specified encoding of NULL values in the input CSV file

A NULL entry is an entry which is considered to be lacking content, specifically in the context of a richer tabular format like an SQL table. Although .csv doesn't explicitly support this characterization for historical reasons, it's a common convention to consider an empty entry containing only white space to be NULL. Therefore, that's the default behavior of the C3R encryption client and it can be customized as needed.

`--csvOutputNULLValue` flag

You can use the `--csvOutputNULLValue` flag to specify custom encodings for NULL entries in the output data when you encrypt data using the C3R encryption client.

The following table summarizes the usage and parameters of this flag.

Usage	Parameters
Optional. Users can specify custom encodings in the generated output file for NULL entries.	User-specified encoding of NULL values in the output CSV file

A NULL entry is an entry which is considered to be lacking content, specifically in the context of a richer tabular format like an SQL table. Although .csv doesn't explicitly support this characterization for historical reasons, it's a common convention to consider an empty entry containing only white space to be NULL. Therefore, that's the default behavior of the C3R encryption client and it can be customized as needed.

`--enableStackTraces` flag

When you encrypt data using the C3R encryption client, use the `--enableStackTraces` flag to provide additional contextual information for error reporting when C3R encounters an error.

AWS doesn't collect errors. If you encounter an error, use the stack trace to troubleshoot the error yourself or send the stack trace to Support for assistance.

The following table summarizes the usage and parameters of this flag.

Usage	Parameters
Optional. Used to provide additional contextual information for error reporting when the C3R encryption client encounters an error.	None

`--dryRun` flag

The encrypt and decrypt C3R encryption client commands include an optional `--dryRun` flag. The flag takes all the user-provided arguments and checks them for validity and consistency.

You can use the `--dryRun` flag to check if your schema file is valid and consistent with its corresponding input file.

The following table summarizes the usage and parameters of this flag.

Usage	Parameters
Optional. Causes the C3R encryption client to parse parameters and check files, but performs no encryption or decryption.	None

`--tempDir` flag

You might want to use a temporary directory because encrypted files can sometimes be larger than non-encrypted files, depending on their settings. Datasets must also be encrypted per collaboration to work correctly.

When you encrypt data using C3R, use the `--tempDir` flag to specify the location where temporary files can be created while processing the input.

The following table summarizes the usage and parameters of this flag.

Usage	Parameters
Users can specify the location where temporary files can be created while processin g the input.	Defaults to the system temporary directory.

Queries with Cryptographic Computing for Clean Rooms

This topic provides information about writing queries that use data tables that have been encrypted using Cryptographic Computing for Clean Rooms.

Topics

- Queries that branch on NULL
- Mapping one source column to multiple target columns
- Using the same data for both JOIN and SELECT queries

Queries that branch on NULL

To have a query branch on a NULL statement means to use syntax like `IF x IS NULL THEN 0 ELSE 1`.

Queries can always branch on NULL statements in cleartext columns.

Queries can branch on NULL statements in sealed columns and fingerprint columns only when the value of the **Preserve NULL values** parameter (`preserveNulls`) is set to `true`.

Queries that violate these constraints might yield incorrect results.

Mapping one source column to multiple target columns

One source column can map to multiple target columns. For example, you might want to both JOIN and SELECT on a column.

For more information, see Using the same data for both JOIN and SELECT queries.

Using the same data for both JOIN and SELECT queries

If the data in a column is not sensitive, it can appear in a cleartext target column, which allows it to be used for any purpose.

If data in a column is sensitive and must be used for both JOIN and SELECT queries, map that source column to two target columns in the output file. One column is encrypted with the `type` as a fingerprint column, and one column is encrypted with the `type` as a sealed column. The interactive schema generation of the C3R encryption client suggests header suffixes of `_fingerprint` and `_sealed`. These header suffixes can be a useful convention for differentiating such columns quickly.

Guidelines for the C3R encryption client

The C3R encryption client is a tool that enables organizations to bring sensitive data together to derive new insights from data analytics. The tool cryptographically limits what can be learned by any party and AWS in the process. Although this is vitally important, the process of securing data cryptographically can add significant overhead both in terms of compute and storage resources. Therefore, it is important to understand the tradeoffs of using each setting and how to optimize settings while still maintaining the desired cryptographic assurances. This topic focuses on the performance implications of different settings in the C3R encryption client and schemas.

All C3R encryption client encryption settings provide different cryptographic assurances. The collaboration-level settings are most secure by default. Enabling additional functionality while creating a collaboration weakens privacy guarantees, allowing activities like frequency analysis to be conducted on the ciphertext. For more information about how these settings are used and what their implications are, see the section called “Cryptographic computing”.

Topics

- Performance implications for column types
- Troubleshooting unanticipated increases in ciphertext size

Performance implications for column types

C3R uses three column types: cleartext, fingerprint, and sealed. Each of these column types provide different cryptographic assurances and have different intended uses. In the following sections, the performance implications of the column type are discussed and the performance impact of each setting.

Topics

- Cleartext columns
- Fingerprint columns
- Sealed columns

Cleartext columns

Cleartext columns are not changed from their original format and not cryptographically processed in any way. This column type can't be configured and does not impact storage or compute performance.

Fingerprint columns

Fingerprint columns are meant to be used for joining data across multiple tables. To this end, the resulting ciphertext size must always be the same. However, these columns are impacted by the collaboration-level settings. Fingerprint columns might have varying degrees of impact on the output file size depending on the cleartext contained in the input.

Topics

- Base overhead for fingerprint columns

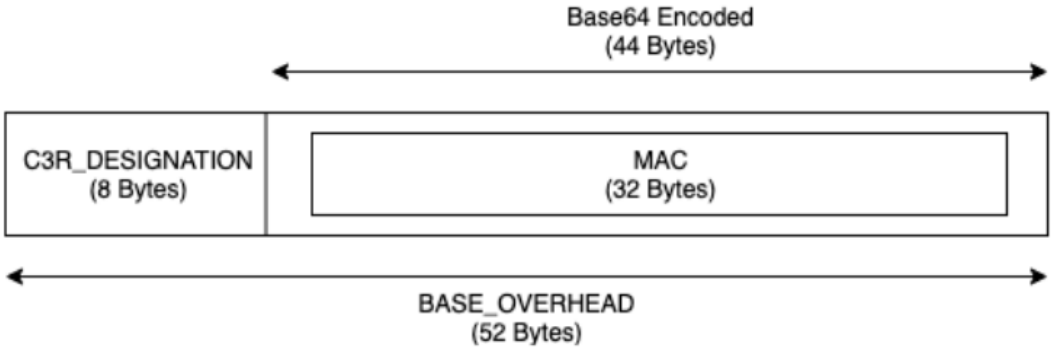
- Collaboration settings for fingerprint columns
- Example data for a fingerprint column
- Troubleshooting fingerprint columns

Base overhead for fingerprint columns

There is a base overhead for fingerprint columns. This overhead is constant and in place of the size of the cleartext bytes.

Data in the fingerprint columns is cryptographically processed through a Hash-based Message Authentication Code (HMAC) function, which turns the data into a 32 byte message authentication code (MAC). This data is then processed through a base64 encoder, adding roughly 33 percent to the byte size. It is pre-pended with an 8 byte C3R designation to designate the type of column that the data belongs to and the client version that produced it. The final result is 52 bytes. This result is then multiplied by the row count to get the total base overhead (use the number of total non-`null` values if `preserveNulls` is set to true).

The following image shows how *`BASE_OVERHEAD = C3R_DESIGNATION + (MAC * 1.33)`*



The output ciphertext in the fingerprint columns will always be 52 bytes. This can be a significant storage decrease if the input cleartext data averages more than 52 bytes (for example, full street addresses). This can be a significant storage increase if the input cleartext data averages less than 52 bytes (for example, customer ages).

Collaboration settings for fingerprint columns

`preserveNulls` setting

When the collaboration-level setting `preserveNulls` is `false` (default), each `null` value is substituted with a unique, random 32 bytes and processed as if it were not `null`. The result is

that each `null` value is now 52 bytes. This can add significant storage requirements for tables that contain very sparse data compared to when this setting is `true` and `null` values are passed through as `null`.

If you don't need the privacy assurances of this setting and prefer to retain `null` values within your datasets, enable the `preserveNulls` setting at the time the collaboration is created. The `preserveNulls` setting can't be changed after the collaboration is created.

Example data for a fingerprint column

The following is an example set of input and output data for a fingerprint column with settings to reproduce. Other collaboration-level settings like `allowCleartext` and `allowDuplicates` don't impact the results and can be set as `true` or `false` if trying to reproduce locally.

Example shared secret: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

Example collaboration ID: `a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`

allowJoinsOnColumnsWithDifferentNames: `True` This setting doesn't impact performance or storage requirements. However, this setting makes column name choice irrelevant when reproducing the values shown in the following tables.

Example 1

Input	`null`
`preserveNulls`	`TRUE`
Output	`null`
Deterministic	`Yes`
Input bytes	0
Output bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`

Output	`01:hmac:3lkFjthvV3IUu6mMvFc1a+XAHwgw/ElmOq4p3Yg25kk=`
Deterministic	No
Input bytes	0
Output bytes	52

Example 3

Input	`empty string`
`preserveNulls`	-
Output	`01:hmac:oKTgi3Gba+eUb3JteSz2EMgXUkF1WgM77UP0Ydw5kPQ=`
Deterministic	`Yes`
Input bytes	0
Output bytes	52

Example 4

Input	`abcdefghijklmnopqrstuvwxyz`
`preserveNulls`	-
Output	`01:hmac:kU/IqwG7FMmzzshr0B9scomE0UJUEE7j9keTctplGww=`
Deterministic	`Yes`
Input bytes	26
Output bytes	52

Example 5

Input	`abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789`
`preserveNulls`	`-`
Output	`01:hmac:ks3htnQbw2vdhCRFF6JNzW5LMndJaHG57uvE26mBtSs=`
Deterministic	`Yes`
Input bytes	62
Output bytes	52

Troubleshooting fingerprint columns

Why is the ciphertext in my fingerprint columns several times greater than the size of the cleartext that went into it?

Ciphertext in a fingerprint column is always 52 bytes in length. If your input data were small (for example, the ages of customers), it will show a significant increase in size. This can also happen if the `preserveNulls` setting is set to `false`.

Why is the ciphertext in my fingerprint columns several times smaller than the size of the cleartext that went into it?

Ciphertext in a fingerprint column is always 52 bytes in length. If your input data were large (for example, the full street addresses of customers), it will show a significant decrease in size.

How do I know if I need the cryptographic assurances provided by `preserveNulls`?

Unfortunately, the answer is that it depends. At a minimum, the the section called “Parameters” should be reviewed for how the `preserveNulls` setting is protecting your data. However, we recommend that you reference your organization's data handling requirements and any contracts applicable to the respective collaboration.

Why do I have to incur the overhead of base64?

To allow for compatibility with tabular file formats such as CSV, base64-encoding is necessary. Although some file formats like Parquet might support binary representations of data, it's important that all participants in a collaboration represent data in the same way to ensure proper query results.

Sealed columns

Sealed columns are meant to be used for transferring data between members of a collaboration. The ciphertext in these columns is non-deterministic and has significant impact on both performance and storage based on how the columns are configured. These columns can be configured individually and often have the greatest impact on the performance of the C3R encryption client and the resulting output file size.

Topics

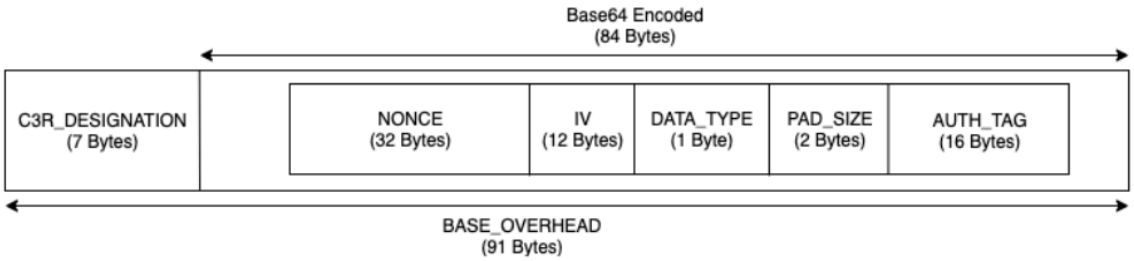
- Base overhead for sealed columns
- Collaboration settings for sealed columns
- Schema settings sealed columns: padding types
- Example data for a sealed column
- Troubleshooting sealed columns

Base overhead for sealed columns

There is a base overhead for sealed columns. This overhead is constant and in addition to the size of the cleartext and padding (if any) bytes.

Before any encryption, data in the sealed columns is pre-pended with a 1 byte character designating what type of data is contained. If padding is selected, the data is then padded and appended with 2 bytes stating the pad size. After these bytes are added, data is cryptographically processed by using AES-GCM and stored with the IV (12 bytes), nonce (32 bytes), and Auth Tag (16 bytes). This data is then processed through a base64 encoder, adding roughly 33 percent to the byte size. The data is pre-pended with a 7 byte C3R designation to designate what type of column the data belongs to and the client version used to produce it. The result is a final base overhead of 91 bytes. This result can then be multiplied by the row count to get the total base overhead (use the number of total non-null values if `preserveNulls` is set to true).

The following image shows how *`BASE_OVERHEAD = C3R_DESIGNATION + ((NONCE + IV + DATA_TYPE + PAD_SIZE + AUTH_TAG) * 1.33)`*



Collaboration settings for sealed columns

`preserveNulls` setting

When the collaboration-level setting `preserveNulls` is `false` (default), each `null` value is unique, random 32 bytes and processed as if it were not `null`. The result is that each `null` value is now 91 bytes (more if padded). This can add significant storage requirements for tables that contain very sparse data compared to when this setting is `true` and `null` values are passed through as `null`.

If you don't need the privacy assurances of this setting and prefer to retain `null` values within your datasets, enable the `preserveNulls` setting at the time the collaboration is created. The `preserveNulls` setting can't be changed after the collaboration is created.

Schema settings sealed columns: padding types

Topics

- Pad type of none
- Pad type of fixed
- Pad type of max

Pad type of none

Selecting a pad type of none doesn't add any padding to the cleartext and adds no additional overhead to the base overhead described earlier. No padding results in the most space-efficient output size. However, it doesn't provide the same privacy assurances as the `fixed` and `max` padding types. This is because the size of the underlying cleartext is discernible from the size of the ciphertext.

Pad type of `fixed`

Selecting a pad type of `fixed` is a privacy-preserving measure to hide the lengths of the data contained within a column. This is done by padding all the cleartext to the provided `pad_length` before it is encrypted. Any data exceeding that size causes the C3R encryption client to fail.

Given that the padding is added to the cleartext before it is encrypted, AES-GCM has a 1-to-1 mapping of cleartext to ciphertext bytes. The base64 encoding will add 33 percent. The additional storage overhead of the padding can be calculated by subtracting the average length of the cleartext from the value of the `pad_length` and multiplying it by 1.33. The result is the average overhead of padding per record. This result can then be multiplied by the number of rows to get the total padding overhead (use the number of total non-`null` values if `preserveNulls` is set to `true`).

*PADDING_OVERHEAD = (PAD_LENGTH - AVG_CLEARTEXT_LENGTH) * 1.33 * ROW_COUNT*

We recommend that you select the minimum `pad_length` that encompasses the largest value in a column. For example, if the largest value is 50 bytes, a `pad_length` of 50 is sufficient. A value larger than that will only add additional storage overhead.

Fixed padding does not add any significant compute overhead.

Pad type of `max`

Selecting a pad type of `max` is a privacy-preserving measure to hide the lengths of the data contained within a column. This is done by padding all the cleartext to the largest value in the column plus the additional `pad_length` before it is encrypted. Generally, `max` padding provides the same assurances as `fixed` padding for a single dataset while allowing for not knowing the largest cleartext value in the column. However, `max` padding might not provide the same privacy assurances as `fixed` padding across updates because the largest value in the individual datasets might differ.

We recommend that you select an additional `pad_length` of 0 when using `max` padding. This length pads all values to be the same size as the largest value in the column. A value larger than that will only add additional storage overhead.

If the largest cleartext value is known for a given column, we recommend that you use the `fixed` pad type instead. Using `fixed` padding creates consistency across updated datasets. Using `max` padding results in each subset of data being padded to the largest value that was in the subset.

Example data for a sealed column

The following is an example set of input and output data for a sealed column with settings to reproduce. Other collaboration-level settings like `allowCleartext`, `allowJoinsOnColumnsWithDifferentNames`, and `allowDuplicates` don't impact the results and can be set as `true` or `false` if trying to reproduce locally. Although these are the basic settings to reproduce, the sealed column is non-deterministic and values will change every time. The goal is to show the bytes in as compared to the bytes out. The example `pad_length` values were chosen intentionally. They show that `fixed` padding results in the same values as `max` padding with the recommended minimum `pad_length` settings or when additional padding is desired.

Example shared secret: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

Example collaboration ID: `a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`

Topics

- Pad type of none
- Pad type of fixed (Example 1)
- Pad type of fixed (Example 2)
- Pad type of max (Example 1)
- Pad type of max (Example 2)

Pad type of none

Example 1

Input	`null`
`preserveNulls`	`TRUE`
Output	`null`
Deterministic	`Yes`
Input bytes	0
Output bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfssGSPbNIJfG3iXmu6cbCUrizuV`
Deterministic	No
Input bytes	0
Output bytes	91

Example 3

Input	`empty string`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfstGSPEM6qR8DWC2PB2GMlX41YK`
Deterministic	No
Input bytes	0
Output bytes	91

Example 4

Input	`abcdefghijklmnopqrstuvwxyz`
`preserveNulls`	-

Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6pkx9sGL5VLDQeHzh6DmPpyWNuI=`
Deterministic	No
Input bytes	26
Output bytes	127

Example 5

Input	`abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6plwtH/8tRFnn2rF91bcB9G4+n8GiRfJNmqdP4/QOQ3cXb/pbvPcnnohrHIGSX54ua+1/JfcVjc=`
Deterministic	No
Input bytes	62
Output bytes	175

Pad type of `fixed` (Example 1)

In this example, `pad_length` is 62 and largest input is 62 bytes.

Example 1

Input	`null`
`preserveNulls`	`TRUE`
Output	`null`
Deterministic	`Yes`
Input bytes	0
Output bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfssGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcoNpATs0GzbnLkor4L+/aSuA=`
Deterministic	No
Input bytes	0
Output bytes	175

Example 3

Input	`empty string`
`preserveNulls`	-

Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfstGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcoLB53l07VZpA6OwkuXu29CA=`
Deterministic	No
Input bytes	0
Output bytes	175

Example 4

Input	`abcdefghijklmnopqrstuvwxyz`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6pkx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcutBAcO+Mb9tuU2KIHH31AWg=`
Deterministic	No
Input bytes	26
Output bytes	175

Example 5

Input	`abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789`
`preserveNulls`	`-`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6plwtH/8tRFnn2rF91bcB9G4+n8GiRfJNmqdP4/QOQ3cXb/pbvPcnnohrHIGSX54ua+1/JfcVjc=`
Deterministic	No
Input bytes	62
Output bytes	175

Pad type of `fixed` (Example 2)

In this example, `pad_length` is 162 and largest input is 62 bytes.

Example 1

Input	`null`
`preserveNulls`	`TRUE`
Output	`null`
Deterministic	`Yes`
Input bytes	0
Output bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfssGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcnkB0xbLWD7zNdAqQGR0rXoSESdW0I0vpNoGcBfv4cJbG0A3h1DvtkSSVc2B80OOGppzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn+8o4WtG/ClipNcjDXvXVtK4vfCohcCA6uwrmwv/xAySX+xcntotL703aBTBb`
Deterministic	No
Input bytes	0
Output bytes	307

Example 3

Input	`empty string`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfstGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcnkB0xbLWD7zNdAqQGR0rXoSESdW0I0vpNoGcBfv4cJbG0A3h1DvtkSSVc2B80OOGp`

	pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwv84lVaT9Yd+6oQx65/+gdVT
Deterministic	No
Input bytes	0
Output bytes	307

Example 4

Input	abcdefghijklmnopqrstuvwxyz
preserveNulls	-
Output	01:enc:bm9uY2UwMTIzNDU2Nzg5 MG5vbmNlMDEyMzQ1Njc4OTBqfRY Z98t5KU6aWfsteEE1GKEPiRzyh0 h7t6OmWMLTWCvO2ckr6pkx9jy48 Fcg1yOPvBqRSZ7oqy1V3UKfYTLE Zb/hCz7oaIneVsrcnkB0xbLWD7z NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwtX5Hnl+WyfO6ks3QMaRDGSf
Deterministic	No
Input bytes	26
Output bytes	307

Example 5

Input	abcdefghijklmnopqrstuvwxyzA BCDEFGHIJKLMNOPQRSTUVWXYZ01 23456789
preserveNulls	-
Output	01:enc:bm9uY2UwMTIzNDU2Nzg5 MG5vbmNlMDEyMzQ1Njc4OTBqfRY Z98t5KU6aWfsteEE1GKEPiRzyh0 h7t6OmWMLTWCvO2ckr6plwtH/8t RFnn2rF91bcB9G4+n8GiRfJNmqd P4/QOQ3cXb/pbvPcnkB0xbLWD7z NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwjkJXQZOgPdeFX9Yr/8alV5i
Deterministic	No
Input bytes	62
Output bytes	307

Pad type of max (Example 1)

In this example, pad_length is 0 and largest input is 62 bytes.

Example 1

Input	null
preserveNulls	TRUE
Output	null
Deterministic	Yes

Input Bytes	0
Output Bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfssGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcoNpATs0GzbnLkor4L+/aSuA=`
Deterministic	No
Input bytes	0
Output bytes	175

Example 3

Input	`empty string`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfstGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcoLB53l07VZpA6OwkuXu29CA=`

Deterministic	No
Input bytes	0
Output bytes	175

Example 4

Input	`abcdefghijklmnopqrstuvwxyz`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6pkx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcutBAcO+Mb9tuU2KIHH31AWg=`
Deterministic	No
Input bytes	26
Output bytes	175

Example 5

Input	`abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789`
`preserveNulls`	-
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfsteEE1GKEPiRzyh0h7t6OmWMLTWCvO2ckr6plwtH/8t`

	RFnn2rF91bcB9G4+n8GiRfJNmqdP4/QOQ3cXb/pbvPcnnohrHIGSX54ua+1/JfcVjc=
Deterministic	No
Input bytes	62
Output bytes	175

Pad type of `max` (Example 2)

In this example, `pad_length` is 100 and largest input is 62 bytes.

Example 1

Input	`null`
`preserveNulls`	`TRUE`
Output	`null`
Deterministic	`Yes`
Input bytes	0
Output bytes	0

Example 2

Input	`null`
`preserveNulls`	`FALSE`
Output	`01:enc:bm9uY2UwMTIzNDU2Nzg5MG5vbmNlMDEyMzQ1Njc4OTBqfRYZ98t5KU6aWfssGSNWfMRp7nSb7SMX2s3JKLOhK1+7r75Tk+Mx9jy48Fcg1yOPvBqRSZ7oqy1V3UKfYTLEZb/hCz7oaIneVsrcnkB0xbLWD7z`

	NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwv/xAySX+xcntotL703aBTBb
Deterministic	No
Input bytes	0
Output bytes	307

Example 3

Input	`empty string`
`preserveNulls`	-
Output	01:enc:bm9uY2UwMTIzNDU2Nzg5 MG5vbmNlMDEyMzQ1Njc4OTBqfRY Z98t5KU6aWfstGSNWfMRp7nSb7S MX2s3JKLOhK1+7r75Tk+Mx9jy48 Fcg1yOPvBqRSZ7oqy1V3UKfYTLE Zb/hCz7oaIneVsrcnkB0xbLWD7z NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwv84lVaT9Yd+6oQx65/+gdVT
Deterministic	No
Input bytes	0
Output bytes	307

Example 4

Input	abcdefghijklmnopqrstuvwxyz
preserveNulls	-
Output	01:enc:bm9uY2UwMTIzNDU2Nzg5 MG5vbmNlMDEyMzQ1Njc4OTBqfRY Z98t5KU6aWfsteEE1GKEPiRzyh0 h7t6OmWMLTWCvO2ckr6pkx9jy48 Fcg1yOPvBqRSZ7oqy1V3UKfYTLE Zb/hCz7oaIneVsrcnkB0xbLWD7z NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwtX5Hnl+WyfO6ks3QMaRDGSf
Deterministic	No
Input bytes	26
Output bytes	307

Example 5

Input	abcdefghijklmnopqrstuvwxyzA BCDEFGHIJKLMNOPQRSTUVWXYZ01 23456789
preserveNulls	-
Output	01:enc:bm9uY2UwMTIzNDU2Nzg5 MG5vbmNlMDEyMzQ1Njc4OTBqfRY Z98t5KU6aWfsteEE1GKEPiRzyh0 h7t6OmWMLTWCvO2ckr6plwtH/8t RFnn2rF91bcB9G4+n8GiRfJNmqd P4/QOQ3cXb/pbvPcnkB0xbLWD7z

	`NdAqQGR0rXoSESdW0I0vpNoGcBf v4cJbG0A3h1DvtkSSVc2B80OOGp pzdDqhrUVN5wFNyn8vgfPMqDaeJk5bn +8o4WtG/ClipNcjDXvXVtK4vfCohcCA6 uwrmwjkJXQZOgPdeFX9Yr/8alV5i`
Deterministic	No
Input bytes	62
Output bytes	307

Troubleshooting sealed columns

Why is the ciphertext in my sealed columns several times greater than the size of the cleartext that went into it?

This depends on several factors. For one, ciphertext in a Cleartext column is always at least 91 bytes in length. If your input data were small (for example, the ages of customers), it will show a significant increase in size. Second, if `preserveNulls` were set to `false` and your input data contained a lot of `null` values, each of those `null` values will have been turned into 91 bytes of ciphertext. Finally, if you use padding, by definition bytes are added to the cleartext data before it is encrypted.

Most of my data in a sealed column is really small, and I need to use padding. Can I just remove the big values and process them separately to save space?

We don't recommend that you remove large values and process them separately. Doing so changes the privacy assurances that the C3R encryption client is providing. As a threat model, assume that an observer can see both encrypted datasets. If the observer sees that one subset of data has a column padded significantly more or less than another subset, they can make inferences on the size of the data in each subset. For example, assume a `fullName` column is padded to a total of 40 bytes in one file and is padded to 800 bytes in another file. An observer might be able to assume that one dataset contains the world’s longest name747 bytes).

Do I need to provide extra padding when using the `max` padding type?

No. When using `max` padding, we recommend that the `pad_length`, also known as the additional padding *beyond* the largest value in the column, is set to 0.

Can I just pick a large `pad_length` when using `fixed` padding to avoid worrying if the largest value will fit?

Yes, but the large pad length is inefficient and uses more storage than necessary. We recommend that you to check to see how large the largest value is and set the `pad_length` to that value.

How do I know if I need the cryptographic assurances provided by `preserveNulls`?

Unfortunately, the answer is that it depends. At a minimum, the Cryptographic Computing for Clean Rooms should be reviewed for how the `preserveNulls` setting is protecting your data. However, we recommend that you reference your organization's data handling requirements and any contracts applicable to the respective collaboration.

Why do I have to incur the overhead of base64?

To allow for compatibility with tabular file formats such as CSV, base64 encoding is necessary. Although some file formats like Parquet might support binary representations of data, it’s important that all participants in a collaboration represent data in the same way to ensure proper query results.

Troubleshooting unanticipated increases in ciphertext size

Let’s say that you encrypted your data, and the size of the resulting data is surprisingly large. The following steps can help you identify where the size increase occurred and what, if any, actions you can take.

Identifying where the size increase occurred

Before you can troubleshoot why your encrypted data is significantly larger than your cleartext data, you must first identify where the increase in size is. Cleartext columns can safely be ignored because they are unchanged. Look at the remaining fingerprint and sealed columns, and choose one that appears significant.

Identifying the reason the size increase occurred

A fingerprint column or a sealed column might contribute to the size increase.

Topics

- Is the size increase coming from a fingerprint column?
- Is the size increase coming from a sealed column?

Is the size increase coming from a fingerprint column?

If the column that's most contributing to the increase in storage is a fingerprint column, this is likely because the cleartext data is small (for example, customer age). Each resulting fingerprint ciphertext is 52 bytes in length. Unfortunately, nothing can be done about this issue on a column-by-column basis. For more information, see Base overhead for fingerprint columns for details about this column, including how it impacts storage requirements.

The other possible cause of size increase in a fingerprint column is the collaboration setting, `preserveNulls`. If the collaboration setting for `preserveNulls` is disabled (the default setting), all `null` values in fingerprint columns will have become 52 bytes of ciphertext. There is nothing that can be done for this in the current collaboration. The `preserveNulls` setting is set at the time a collaboration is created and all collaborators must use the same setting to ensure correct query results. For more information about the `preserveNulls` setting and how enabling it impacts the privacy assurances of your data, see the section called "Cryptographic computing".

Is the size increase coming from a sealed column?

If the column that's most contributing to the increase in storage is a sealed column, there are a few details that could contribute to the size increase.

If the cleartext data is small (for example, customer age), each resulting sealed ciphertext is at least 91 bytes in length. Unfortunately, nothing can be done about this issue. For more information, see Base overhead for sealed columns for details about this column, including how it impacts storage requirements.

The second primary cause for storage increase in sealed columns is padding. Padding adds extra bytes to the cleartext before it's encrypted to hide the size of individual values in a dataset. We recommend that you set padding to the minimum possible value for your dataset. At a minimum, `pad_length` for `fixed` padding must be set to encompass the largest possible value in the column. Any higher setting than that doesn't add additional privacy assurances. For example, if you know the largest possible value in a column can be 50 bytes, we recommend that you set the `pad_length` to 50 bytes. However, if the sealed column is using `max` padding, we recommend that you set the `pad_length` to 0 bytes. This is because `max` padding is referring to the *additional* padding beyond the largest value in the column.

The final possible cause of size increase in a sealed column is the collaboration setting, `preserveNulls`. If the collaboration setting for `preserveNulls` is disabled (the default setting), all `null` values in sealed columns will have become 91 bytes of ciphertext. There is nothing that

can be done for this in the current collaboration. The `preserveNulls` setting is set at the time a collaboration is created, and all collaborators must use the same setting to ensure correct query results. For more information about this setting does and how enabling it impacts the privacy assurances of your data, see the section called “Cryptographic computing”.

Setting up AWS Clean Rooms

The following topics explain how to set up AWS Clean Rooms.

Topics

- Sign up for an AWS account
- Set up service roles for AWS Clean Rooms
- Set up service roles for AWS Clean Rooms ML

Sign up for an AWS account

To get started with AWS, you need an AWS account. For information about creating an AWS account, see Getting started with an AWS account in the *AWS Account Management Reference Guide*.

Set up service roles for AWS Clean Rooms

The following sections describe the roles needed to perform each task.

Topics

- Create an IAM role for a collaboration member
- Create a service role to read data from Amazon S3
- Create a service role to read data from Amazon Athena
- Create a service role to read data from Snowflake
- Create a service role to read code from an S3 bucket (PySpark analysis template role)
- Create a service role to write results of a PySpark job
- Create a service role to receive results

Create an IAM role for a collaboration member

A member is an AWS customer who is a participant in a collaboration.

To create an IAM role for a collaboration member

1. Follow the Creating a role to delegate permissions to an IAM user procedure in the *AWS Identity and Access Management User Guide*.
2. For the **Create policy** step, select the **JSON** tab in the **Policy editor**, and then add policies depending on the abilities granted to the collaboration member.

   AWS Clean Rooms offers the following managed policies based on common use cases.

If you want to ...	Then use ...
View the resources and metadata	AWS managed policy: `AWSCleanRoomsReadOnlyAccess`
Query	AWS managed policy: `AWSCleanRoomsFullAccess`
Query and run jobs	AWS managed policy: `AWSCleanRoomsFullAccess`
Query and receive results	AWS managed policy: `AWSCleanRoomsFullAccess`
Manage collaboration resources but do not query	AWS managed policy: `AWSCleanRoomsFullAccessNoQuerying`

   For information about the different managed policies offered by AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms,

Create a service role to read data from Amazon S3

AWS Clean Rooms uses a service role to read the data from Amazon S3.

There are two ways to create this service role.

- If you have the necessary IAM permissions to create a service role, use the AWS Clean Rooms console to create a service role.

- If you don't have `iam:CreateRole`, `iam:CreatePolicy` and `iam:AttachRolePolicy` permissions or want to create the IAM roles manually, do one of the following:
  - Use the following procedure to create a service role using custom trust policies.
  - Ask your administrator to create the service role using the following procedure.

> **Note**
>
> You or your IAM administrator should follow this procedure only if you don't have the necessary permissions to create a service role using the AWS Clean Rooms console.

To create a service role to read data from Amazon S3 using custom trust policies

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

   > **Note**
   >
   > If you want to help ensure that the role is used only in the context of a certain collaboration membership, you can scope down the trust policy further. For more information, see Cross-service confused deputy prevention.

   JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyForCleanRoomsService",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Use the following permissions policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Amazon S3 data. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. For instance, if you have set up a custom KMS key for your Amazon S3 data, you may need to amend this policy with additional AWS Key Management Service (AWS KMS) permissions.
> Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "NecessaryGluePermissions",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:database/databaseName",
```

```
                "arn:aws:glue:us-
east-1:111122223333:table/databaseName/tableName",
                "arn:aws:glue:us-east-1:111122223333:catalog"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetSchema",
                "glue:GetSchemaVersion"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "NecessaryS3BucketPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::bucket"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "444455556666"
                    ]
                }
            }
        },
        {
            "Sid": "NecessaryS3ObjectPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket/prefix/*"
            ],
            "Condition": {
                "StringEquals": {
```

```
                    "s3:ResourceAccount": [
                        "444455556666"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> This policy references two different AWS account IDs to support a AWS Clean Rooms collaboration where data catalog metadata and actual data storage are managed by different parties:
>
> - **111122223333** - This is the account that owns the AWS Glue Data Catalog resources (databases, tables, and catalog). The first statement grants permissions to access table schemas, partition information, and metadata from this account's AWS Glue catalog.
> - **444455556666** - This is the account that owns the Amazon S3 bucket containing the actual data files. The Amazon S3 permissions (statements 3 and 4) are restricted to buckets owned by this account through the `s3:ResourceAccount` condition.
>
> This configuration supports common enterprise data architectures where one team manages the data catalog and schema definitions while another team owns the underlying data storage infrastructure. The `s3:ResourceAccount` condition provides an additional security layer by ensuring Amazon S3 operations only work on buckets owned by the designated account.

4. Replace each *`placeholder`* with your own information.
5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Create a service role to read data from Amazon Athena

AWS Clean Rooms uses a service role to read the data from Amazon Athena.

To create a service role to read data from Athena using custom trust policies

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> If you want to help ensure that the role is used only in the context of a certain collaboration membership, you can scope down the trust policy further. For more information, see Cross-service confused deputy prevention.

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyForCleanRoomsService",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
     ]
}
```

3. Use the following permissions policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Athena data. However, you might need to modify this

policy depending on how you've set up your Amazon S3 data. For instance, if you've already set up a custom KMS key for your Amazon S3 data, you may need to amend this policy with additional AWS KMS permissions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:GetWorkGroup",
                "athena:GetTableMetadata",
                "athena:GetDataCatalog",
                "athena:StartQueryExecution",
                "athena:GetQueryExecution",
                "athena:GetQueryResults"
            ],
            "Resource": [
                "arn:aws:athena:region:accountId:workgroup/workgroup",
                "arn:aws:athena:region:accountId:datacatalog/federatedCatalogName"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetTable",
                "glue:GetCatalog"
            ],
            "Resource": [
                "arn:aws:glue:region:accountId:catalog",
                "arn:aws:glue:region:accountId:catalog/federatedCatalogName",

 "arn:aws:glue:region:accountId:database/federatedCatalogName/databaseName",

 "arn:aws:glue:region:accountId:table/federatedCatalogName/databaseName/tableName"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:AbortMultipartUpload",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::athenaResultsBucket",
                "arn:aws:s3:::athenaResultsBucket/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "accountId"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "lakeformation:GetDataAccess",
            "Resource": "*"
        }
    ]
}
```

4. Replace each *placeholder* with your own information.
5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Set up Lake Formation permissions

If you query resources protected with Lake Formation permissions, the service role must have **Select** and **Describe** access permissions on the table/view/catalog and **Describe** permissions on the AWS Glue database.

For more information, see:

- Use Athena to query data registered with AWS Lake Formation in the *Amazon Athena User Guide*
- Onboarding to Lake Formation permissions in the *AWS Lake Formation Developer Guide*

Create a service role to read data from Snowflake

AWS Clean Rooms uses a service role to retrieve your credentials for Snowflake to read your data from this source.

There are two ways to create this service role:

- If you have the necessary IAM permissions to create a service role, use the AWS Clean Rooms console to create a service role.
- If you don't have `iam:CreateRole`, `iam:CreatePolicy` and `iam:AttachRolePolicy` permissions or want to create the IAM roles manually, do one of the following:
  - Use the following procedure to create a service role using custom trust policies.
  - Ask your administrator to create the service role using the following procedure.

> **Note**
>
> You or your IAM administrator should follow this procedure only if you don't have the necessary permissions to create a service role using the AWS Clean Rooms console.

To create a service role to read data from Snowflake using custom trust policies

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

   > **Note**
   >
   > If you want to help ensure that the role is used only in the context of a certain collaboration membership, you can scope down the trust policy further. For more information, see Cross-service confused deputy prevention.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIfSourceArnMatches",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ForAnyValue:ArnEquals": {
                    "aws:SourceArn": [
                        "arn:aws:cleanrooms:us-
east-1:111122223333:membership/membershipId",
                        "arn:aws:cleanrooms:us-
east-1:444455556666:membership/queryRunnerMembershipId"
                    ]
                }
            }
        }
    ]
}
```

Note

This trust policy references two different AWS account IDs to support a AWS Clean Rooms collaboration where query execution responsibilities are distributed across multiple parties:

- **111122223333** - This is the account that contains a membership participating in the collaboration. This membership may own data tables, analysis rules, or other collaboration resources that require role access.
- **444455556666** - This is the account that contains the membership responsible for running queries (the "query runner"). This membership executes protected queries and needs to assume this role to access the necessary compute and data resources.

This configuration enables scenarios where one party provides data or analysis templates while another party runs the actual queries. Both roles require different but complementary permissions through the same execution role. The `aws:SourceArn` condition ensures that only AWS Clean Rooms operations originating from these two specific memberships can assume the role, maintaining security while supporting the distributed job execution and result management workflow.

3. Use one of the following permissions policies according to the Creating a role using custom trust policies (console) procedure.

Permission policy for secrets encrypted with a customer-owned KMS key

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secretIdentifier",
            "Effect": "Allow"
        },
        {
            "Sid": "AllowDecryptViaSecretsManagerForKey",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-
east-1:444455556666:key/keyIdentifier",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.us-
east-1.amazonaws.com",
                    "kms:EncryptionContext:SecretARN":
 "arn:aws:secretsmanager:us-east-1:111122223333:secret:secretIdentifier"
                }
            }
        }
    ]
```

```
}
```

> **Note**
>
> This policy references two different AWS account IDs to support a cross-account secrets management scenario:
>
> - **111122223333** - This is the account that owns and stores the secret in . The first statement grants permission to retrieve the secret value from this account.
> - **444455556666** - This is the account that owns the AWS KMS key used to encrypt the secret. The second statement grants permission to decrypt the secret using the AWS KMS key from this account.
>
> This configuration is common in enterprise environments where:
>
> - Secrets are centrally managed in one account (Account 1)
> - Encryption keys are managed by a separate security or shared services account (Account 2)
> - The AWS KMS key policy in Account 2 must also allow the service in Account 1 to use the key for encryption/decryption operations
>
> The `kms:EncryptionContext:SecretARN` condition ensures that the AWS KMS key can only be used to decrypt this specific secret, providing an additional layer of security for cross-account access.

Permission policy for secrets encrypted with an AWS managed key

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "secretsmanager:GetSecretValue",
```

```
            "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secretIdentifier",
            "Effect": "Allow"
        }
    ]
}
```

4. Replace each *placeholder* with your own information.
5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Create a service role to read code from an S3 bucket (PySpark analysis template role)

AWS Clean Rooms uses a service role to read code from a collaboration member's specified S3 bucket when using a PySpark analysis template.

To create a service role to read code from an S3 bucket

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ForAnyValue:ArnEquals": {
                    "aws:SourceArn": [
```

```
                        "arn:aws:cleanrooms:us-
east-1:111122223333:membership/jobRunnerMembershipId",
                        "arn:aws:cleanrooms:us-
east-1:444455556666:membership/analysisTemplateOwnerMembershipId"
                    ]
                }
            }
        }
    ]
}
```

Note

This trust policy references two different AWS account IDs to support a multi-party AWS Clean Rooms collaboration scenario:

- **111122223333** - This is the account that contains the membership responsible for running queries (the "job runner"). This membership executes the analysis jobs and needs to assume this role to access the necessary resources.
- **444455556666** - This is the account that owns the analysis template and its associated membership (the "analysis template owner"). This membership defines what queries can be run and also needs to assume this role to manage and execute the analysis.

This configuration is typical in AWS Clean Rooms collaborations where multiple parties participate in the same collaboration, each with their own AWS account and membership. Both the query executor and the analysis template owner need access to shared resources. The `aws:SourceArn` condition ensures that only AWS Clean Rooms operations originating from these two specific memberships can assume the role, providing precise access control for the multi-party collaboration.

3. Use the following permissions policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> The following example policy supports the permissions needed to read your code from Amazon S3. However, you might need to modify this policy depending on how you've set up your S3 data.
>
> Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": ["arn:aws:s3:::s3Path"],
            "Condition":{
                "StringEquals":{
                    "s3:ResourceAccount":[
                        "s3BucketOwnerAccountId"
                    ]
                }
            }
        }
    ]
}
```

4. Replace each *placeholder* with your own information:

   - *s3Path* – The S3 bucket location of your code.
   - *s3BucketOwnerAccountId* – The AWS account ID of the S3 bucket owner.
   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *jobRunnerAccountId* – The AWS account ID of the member who can run queries and jobs.

- *jobRunnerMembershipId* – The **Membership ID** of the member who can query and run jobs. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
- *analysisTemplateAccountId* – The AWS account ID of the analysis template.
- *analysisTemplateOwnerMembershipId* – The **Membership ID** of the member who owns the analysis template. The **Membership ID** can be found on the **Details** tab of the collaboration.

5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Create a service role to write results of a PySpark job

AWS Clean Rooms uses a service role to write the results of a PySpark job to a specified S3 bucket.

To create a service role to write results of a PySpark job

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ForAnyValue:ArnEquals": {
                    "aws:SourceArn": [
```

```
                        "arn:aws:cleanrooms:us-
east-1:111122223333:membership/jobRunnerMembershipId",
                        "arn:aws:cleanrooms:us-
east-1:444455556666:membership/rrMembershipId"
                    ]
                }
            }
        }
    ]
}
```

Note

This trust policy references two different AWS account IDs to support a AWS Clean Rooms collaboration with distinct operational roles:

- **111122223333** - This is the account that contains the membership responsible for running analysis jobs (the "job runner"). This membership executes the computational workloads and needs to assume this role to access processing resources.
- **444455556666** - This is the account that contains the membership with result receiver (RR) responsibilities. This membership is authorized to receive and access the output of analysis jobs, and needs role access to write results to designated locations.

This configuration enables AWS Clean Rooms scenarios where one party runs the computational analysis while another party receives and manages the results. Both roles require different but complementary permissions through the same execution role. The `aws:SourceArn` condition ensures that only AWS Clean Rooms operations originating from these two specific memberships can assume the role, maintaining security while supporting the distributed job execution and result management workflow.

3. Use the following permissions policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> The following example policy supports the permissions needed to write to Amazon S3. However, you might need to modify this policy depending on how you've set up S3. Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::bucket/optionalPrefix/*",
            "Condition":{
                "StringEquals":{
                    "s3:ResourceAccount":[
                        "s3BucketOwnerAccountId"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::bucket",
            "Condition":{
                "StringEquals":{
                    "s3:ResourceAccount":[
                        "s3BucketOwnerAccountId"
                    ]
                }
```

```
            }
        }
    ]
}
```

4. Replace each *placeholder* with your own information:

   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *jobRunnerAccountId* – The AWS account ID in which the S3 bucket is located.
   - *jobRunnerMembershipId* – The **Membership ID** of the member who can query and run jobs. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
   - *rrAccountId* – The AWS account ID in which the S3 bucket is located.
   - *rrMembershipId* – The **Membership ID** of the member who can receive results. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
   - *bucket* – The name and location of the S3 bucket.
   - *optionalPrefix* – An optional prefix if you want to save your results under a specific S3 prefix.
   - *s3BucketOwnerAccountId* – The AWS account ID of the S3 bucket owner.
5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Create a service role to receive results

> **Note**
>
> If you are the member who can only receive results (in the console, **Your member abilities** is only **Receive results**), follow this procedure.
> If you are a member who can both query and receive results (in the console, **Your member abilities** is both **Query** and **Receive results**), you can skip this procedure.

For collaboration members who can only receive results, AWS Clean Rooms uses a service role to write results of the queried data in the collaboration to the specified S3 bucket.

There are two ways to create this service role:

- If you have the necessary IAM permissions to create a service role, use the AWS Clean Rooms console to create a service role.
- If you don't have `iam:CreateRole`, `iam:CreatePolicy` and `iam:AttachRolePolicy` permissions or want to create the IAM roles manually, do one of the following:
  - Use the following procedure to create a service role using custom trust policies.
  - Ask your administrator to create the service role using the following procedure.

> **Note**
>
> You or your IAM administrator should follow this procedure only if you don't have the necessary permissions to create a service role using the AWS Clean Rooms console.

To create a service role to receive results using custom trust policies

1. Create a role using custom trust policies. For more information, see the Creating a role using custom trust policies (console) procedure in the *AWS Identity and Access Management User Guide*.
2. Use the following custom trust policy according to the Creating a role using custom trust policies (console) procedure.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIfExternalIdMatches",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
```

```
            "Condition": {
                "ArnLike": {
                    "sts:ExternalId":
 "arn:aws:*:region:*:dbuser:*/a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa*"
                }
            }
        },
        {
            "Sid": "AllowIfSourceArnMatches",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ForAnyValue:ArnEquals": {
                    "aws:SourceArn": [
                        "arn:aws:cleanrooms:us-
east-1:555555555555:membership/a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa"
                    ]

                }
            }
        }
    ]
}
```

3. Use the following permissions policy according to the Creating a role using custom trust policies (console) procedure.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Amazon S3 data. However, you might need to modify this policy depending on how you've set up your S3 data.
> Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::bucket_name"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount":"accountId"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket_name/optional_key_prefix/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount":"accountId"
                }
            }
        }
    ]
}
```

4. Replace each *placeholder* with your own information:

   - *region* – The name of the AWS Region. For example, **us-east-1**.

- *a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa* – The **Membership ID** of the member who can query. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
- *arn:aws:cleanrooms:us-east-1:555555555555:membership/a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa* – The single **Membership ARN** of the member who can query. The **Membership ARN** can be found on the **Details** tab of the collaboration. This ensures AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
- *bucket_name* – The **Amazon Resource Name (ARN)** of the S3 bucket. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.
- *accountId* – The AWS account ID in which the S3 bucket is located.

  bucket_name/optional_key_prefix – The **Amazon Resource Name (ARN)** of the results destination in Amazon S3. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.

5. Continue to follow the Creating a role using custom trust policies (console) procedure to create the role.

Set up service roles for AWS Clean Rooms ML

The roles needed to perform lookalike modeling differ from those needed to use a custom model. The following sections describe the roles needed to perform each task.

Topics

- Set up service roles for lookalike modeling
- Set up service roles for custom modeling

Set up service roles for lookalike modeling

Topics

- Create a service role to read training data
- Create a service role to write a lookalike segment
- Create a service role to read seed data

Create a service role to read training data

AWS Clean Rooms uses a service role to read training data. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

To create a service role to train a dataset

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.
3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Amazon S3 data. However, you might need to modify this policy depending on how you've set up your S3 data. This policy doesn't include a KMS key to decrypt data.
> Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:GetPartitions",
                "glue:GetPartition",
                "glue:BatchGetPartition",
```

```
                "glue:GetUserDefinedFunctions"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:database/databases",
                "arn:aws:glue:us-east-1:111122223333:table/databases/tables",
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/default"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:database/default"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::bucket"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "111122223333"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucketFolders/*"
            ],
            "Condition": {
```

```
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "111122223333"
                    ]
                }
            }
        }
    ]
}
```

If you need to use a KMS key to decrypt data, add this AWS KMS statement to the previous template:

```
{
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
            ],
            "Resource": [
                "arn:aws:kms:region:accountId:key/keyId"
            ],
            "Condition": {
                "ArnLike": {
                        "kms:EncryptionContext:aws:s3:arn":
 "arn:aws:s3:::bucketFolders*"
                }
            }
        }
    ]
}
```

5. Replace each *placeholder* with your own information:

   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *accountId* – The AWS account ID in which the S3 bucket is located.
   - *database/databases*, *table/databases/tables*, *catalog*, and *database/default* – The location of the training data that AWS Clean Rooms needs to access.
   - *bucket* – The **Amazon Resource Name (ARN)** of the S3 bucket. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.

- *bucketFolders* – The name of specific folders in the S3 bucket that AWS Clean Rooms needs to access.

6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.

9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.

10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRole",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:SourceAccount": ["111122223333"]
                },
                "ArnLikeIfExists": {
                    "aws:SourceArn": "arn:aws:cleanrooms-ml:us-
east-1:111122223333:training-dataset/*"
                }
            }
        }
    ]
}
```

The `SourceAccount` is always your AWS account. The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

`accountId` is the ID of AWS account that contains the training data.

13. Choose **Next** and under **Add permissions**, enter the name of the policy you just created. (You might need to reload the page.)
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Create a service role to write a lookalike segment

AWS Clean Rooms uses a service role to write lookalike segments to a bucket. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

To create a service role to write a lookalike segment

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.

3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Amazon S3 data. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::buckets"
            ],
            "Condition":{
                "StringEquals":{
                    "s3:ResourceAccount":[
                        "accountId"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::bucketFolders/*"
            ],
            "Condition":{
                "StringEquals":{
                    "s3:ResourceAccount":[
                        "accountId"
                    ]
                }
            }
        }
  ]
}
```

If you need to use a KMS key to encrypt data, add this AWS KMS statement to the template:

```
{
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:GenerateDataKey*",
                "kms:ReEncrypt*",
            ],
            "Resource": [
                "arn:aws:kms:region:accountId:key/keyId"
            ],
            "Condition": {
                "ArnLike": {
                        "kms:EncryptionContext:aws:s3:arn":
 "arn:aws:s3:::bucketFolders*"
                }
            }
        }
  ]
}
```

5. Replace each *placeholder* with your own information:

   - *buckets* – The **Amazon Resource Name (ARN)** of the S3 bucket. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.
   - *accountId* – The AWS account ID in which the S3 bucket is located.

- *bucketFolders* – The name of specific folders in the S3 bucket that AWS Clean Rooms needs to access.
- *region* – The name of the AWS Region. For example, **us-east-1**.
- *keyId* – The KMS key needed to encrypt your data.

6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRole",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:SourceAccount": ["111122223333"]

                },
                "ArnLikeIfExists": {
                    "aws:SourceArn": "arn:aws:cleanrooms-ml:us-
east-1:111122223333:configured-audience-model/*"
                }
```

```
            }
        }
    ]
}
```

The `SourceAccount` is always your AWS account. The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Create a service role to read seed data

AWS Clean Rooms uses a service role to read seed data. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

To create a service role to read seed data that is stored in an S3 bucket.

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.

3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste one of the following policies.

> **Note**
>
> The following example policy supports the permissions needed to read AWS Glue metadata and its corresponding Amazon S3 data. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your AWS Glue resources and underlying Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::buckets"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "accountId"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::bucketFolders/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "accountId"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> The following example policy supports the permissions needed to read the results of an SQL query and use that as the input data. However, you might need to modify this policy depending on how your query is structured. This policy doesn't include a KMS key to decrypt data.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCleanRoomsStartQuery",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:GetCollaborationAnalysisTemplate",
                "cleanrooms:GetSchema",
                "cleanrooms:StartProtectedQuery"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowCleanRoomsGetAndUpdateQuery",
            "Effect": "Allow",
```

```
            "Action": [
                "cleanrooms:GetProtectedQuery",
                "cleanrooms:UpdateProtectedQuery"
            ],
            "Resource": [
                "arn:aws:cleanrooms:us-
east-1:111122223333:membership/queryRunnerMembershipId"
            ]
        }
    ]
}
```

If you need to use a KMS key to decrypt data, add this AWS KMS statement to the template:

```
{
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:region:accountId:key/keyId"
            ],
            "Condition": {
                "ArnLike": {
                        "kms:EncryptionContext:aws:s3:arn":
 "arn:aws:s3:::bucketFolders*"
                }
            }
        }
  ]
}
```

5. Replace each *placeholder* with your own information:

   - *buckets* – The **Amazon Resource Name (ARN)** of the S3 bucket. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.
   - *accountId* – The AWS account ID in which the S3 bucket is located.
   - *bucketFolders* – The name of specific folders in the S3 bucket that AWS Clean Rooms needs to access.
   - *region* – The name of the AWS Region. For example, **us-east-1**.

- *queryRunnerAccountId* – The AWS account ID of the account that will run queries.
- *queryRunnerMembershipId* – The **Membership ID** of the member who can query. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
- *keyId* – The KMS key needed to encrypt your data.

6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRole",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:SourceAccount": ["111122223333"]

                },
                "ArnLikeIfExists": {
```

```
                    "aws:SourceArn": "arn:aws:cleanrooms-ml:us-
east-1:111122223333:audience-generation-job/*"
                }
            }
        }
    ]
}
```

The `SourceAccount` is always your AWS account. The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Set up service roles for custom modeling

Topics

- Create a service role for custom ML modeling - ML Configuration
- Create a service role to provide a custom ML model
- Create a service role to query a dataset

- Create a service role to create a configured table association

Create a service role for custom ML modeling - ML Configuration

AWS Clean Rooms uses a service role to control who can create a custom ML configuration. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

This role allows you to use the PutMLConfiguration action.

To create a service role to allow creation of a custom ML configuration

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.
3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the permissions needed to access and write data to an S3 bucket and to publish CloudWatch metrics. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ObjectWriteForExport",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::bucket/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:ResourceAccount": [
                        "111122223333"
                    ]
                }
            }
        },
        {
            "Sid": "AllowS3KMSEncryptForExport",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:GenerateDataKey*"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/keyId"
            ],
            "Condition": {
                "StringLike": {
                    "kms:EncryptionContext:aws:s3:arn":
"arn:aws:s3:::bucket*"
                }
            }
        },
        {
            "Sid": "AllowCloudWatchMetricsPublishingForTrainingJobs",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringLike": {
                    "cloudwatch:namespace": "/aws/cleanroomsml/*"
                }
            }
        },
        {
            "Sid": "AllowCloudWatchLogsPublishingForTrainingOrInferenceJobs",
            "Effect": "Allow",
            "Action": [
```

```
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
cleanroomsml/*"
            ]
        }
    ]
}
```

5. Replace each *placeholder* with your own information:

   - *bucket* – The **Amazon Resource Name (ARN)** of the S3 bucket. The **Amazon Resource Name (ARN)** can be found on the **Properties** tab of the bucket in Amazon S3.
   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *accountId* – The AWS account ID in which the S3 bucket is located.
   - *keyId* – The KMS key needed to encrypt your data.
6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cleanrooms:us-
east-1:111122223333:membership/membershipID"
                }
            }
        }
    ]
}
```

The `SourceAccount` is always your AWS account. The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

a. Review **Select trusted entities**, and edit if necessary.
b. Review the permissions in **Add permissions**, and edit if necessary.
c. Review the **Tags**, and add tags if necessary.
d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Create a service role to provide a custom ML model

AWS Clean Rooms uses a service role to control who can create a custom ML model algorithm. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

This role allows you to use the CreateConfiguredModelAlgorithm action.

To create a service role to allow a member to provide a custom ML model

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.
3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the permissions needed to retrieve the docker image that contains the model algorithm. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowECRImageDownloadForTrainingAndInferenceJobs",
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:BatchCheckLayerAvailability",
```

```
                "ecr:GetDownloadUrlForLayer"
            ],
            "Resource": "arn:aws:ecr:us-
east-1:111122223333:repository/repoName"
        }
    ]
}
```

5. Replace each *placeholder* with your own information:
   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *accountId* – The AWS account ID in which the S3 bucket is located.
   - *repoName* – The name of the repository that contains your data.
6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

The `SourceAccount` is always your AWS account The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Create a service role to query a dataset

AWS Clean Rooms uses a service role to control who can query a dataset that will be used for custom ML modeling. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

This role allows you to use the CreateMLInputChannel action.

To create a service role to allow a member to query a dataset

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.

2. Under **Access management**, choose **Policies**.
3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the permissions needed to query a dataset that will be used for custom ML modeling. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCleanRoomsStartQueryForMLInputChannel",
            "Effect": "Allow",
            "Action": "cleanrooms:StartProtectedQuery",
            "Resource": "*"
        },
        {
            "Sid":
 "AllowCleanroomsGetSchemaAndGetAnalysisTemplateForMLInputChannel",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:GetSchema",
                "cleanrooms:GetCollaborationAnalysisTemplate"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowCleanRoomsGetAndUpdateQueryForMLInputChannel",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:GetProtectedQuery",
```

```
                "cleanrooms:UpdateProtectedQuery"
            ],
            "Resource": [
                "arn:aws:cleanrooms:us-
east-1:111122223333:membership/queryRunnerMembershipId"
            ]
        }
    ]
}
```

5. Replace each *placeholder* with your own information:

   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *queryRunnerAccountId* – The AWS account ID of the account that will run the queries.
   - *queryRunnerMembershipId* – The **Membership ID** of the member who can query. The **Membership ID** can be found on the **Details** tab of the collaboration. This ensures that AWS Clean Rooms is assuming the role only when this member runs the analysis in this collaboration.
6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.
11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

The `SourceAccount` is always your AWS account The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.

13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

> **Note**
>
> The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Create a service role to create a configured table association

AWS Clean Rooms uses a service role to control who can create a configured table association. You can create this role using the console if you have the necessary IAM permissions. If you don't have `CreateRole` permissions, ask your administrator to create the service role.

This role allows you to use the CreateConfiguredTableAssociation action.

To create a service role to allow creation of a configured table association

1. Sign in to the IAM console (https://console.aws.amazon.com/iam/) with your administrator account.
2. Under **Access management**, choose **Policies**.
3. Choose **Create policy**.
4. In the **Policy editor**, select the **JSON** tab, and then copy and paste the following policy.

> **Note**
>
> The following example policy supports the creation of a configured table association. However, you might need to modify this policy depending on how you've set up your Amazon S3 data. This policy doesn't include a KMS key to decrypt data.
> Your Amazon S3 resources must be in the same AWS Region as the AWS Clean Rooms collaboration.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/KMS-key-ID",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::bucket-name",
            "Effect": "Allow"
        },
        {
```

```
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::bucket-name/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:GetPartitions",
                "glue:GetPartition",
                "glue:BatchGetPartition"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/Glue database
 name",
                "arn:aws:glue:us-east-1:111122223333:table/Glue database
 name/Glue table name"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "glue:GetSchema",
                "glue:GetSchemaVersion"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Replace Placeholder Resource ARNs

When using this policy, you must replace the placeholder resource identifiers with the actual ARNs of your resources:

- **AWS KMS Key Resource**: Replace *KMS-key-ID* with the actual AWS KMS key ID that encrypts your Amazon S3 data. The key must be in the same account (111122223333) that owns the AWS Glue catalog resources.

- **Amazon S3 Bucket Resources**: Replace *bucket-name* with the actual name of the Amazon S3 bucket that contains your AWS Glue table data. Note that Amazon S3 bucket ARNs don't include account IDs since bucket names are globally unique.
- **AWS Glue Resources**: Replace the following placeholders with your actual resource names:
  - *Glue database name* - The name of your AWS Glue database
  - *Glue table name* - The name of your AWS Glue table

All AWS Glue resources (catalog, database, and table) must be in the same AWS account (111122223333) to ensure consistent access permissions. This account should be the same one that owns the AWS KMS key used for data encryption, creating a unified security boundary for your AWS Clean Rooms data resources.

5. Replace each *placeholder* with your own information:
   - *KMS key used to encrypt the Amazon S3 data* – The KMS key that was used to encrypt the Amazon S3 data. In order to decrypt the data, you need to provide the same KMS key that was used to encrypt the data.
   - *Amazon S3 bucket of AWS Glue table* – The name of the Amazon S3 bucket that contains the AWS Glue table that contains your data.
   - *region* – The name of the AWS Region. For example, **us-east-1**.
   - *accountId* – The AWS account ID of the account that owns the data.
   - *AWS Glue database name* – The name of the AWS Glue database that contains your data.
   - *AWS Glue table name* – The name of the AWS Glue table that contains your data.
6. Choose **Next**.
7. For **Review and create**, enter a **Policy name** and **Description**, and review the **Summary**.
8. Choose **Create policy**.

   You have created a policy for AWS Clean Rooms.
9. Under **Access management**, choose **Roles**.

   With **Roles**, you can create short-term credentials, which is recommended for increased security. You can also choose **Users** to create long-term credentials.
10. Choose **Create role**.

11. In the **Create role** wizard, for **Trusted entity type**, choose **Custom trust policy**.
12. Copy and paste the following custom trust policy into the JSON editor.

    JSON

    ```
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "cleanrooms-ml.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }
    ```

    The `SourceAccount` is always your AWS account The `SourceArn` can be limited to a specific training dataset, but only after that dataset is created. Because you don't yet know the training dataset ARN, the wildcard is specified here.
13. Choose **Next**.
14. Select the check box next to the name of the policy you created, and then choose **Next**.
15. For **Name, review, and create**, enter the **Role name** and **Description**.

    > **Note**
    >
    > The **Role name** must match the pattern in the `passRole` permissions granted to the member who can query and receive results and member roles.

    a. Review **Select trusted entities**, and edit if necessary.
    b. Review the permissions in **Add permissions**, and edit if necessary.
    c. Review the **Tags**, and add tags if necessary.
    d. Choose **Create role**.

You have created the service role for AWS Clean Rooms.

Collaborations and memberships in AWS Clean Rooms

A *collaboration* is a secure logical boundary in AWS Clean Rooms in which members can perform analysis on configured tables.

Any member in AWS Clean Rooms can create a collaboration.

The collaboration creator can designate a single member to analyze configured tables and receive results. However, the collaboration creator might want to prevent the member who can run the analysis from having access to the query results. In that case, the collaboration creator can designate one member to who can query or one member who can run queries and jobs and another member who can receive results.

In most cases, the member who can query or the member who can query and run jobs is also the member paying for compute costs. However, the collaboration creator can configure a different member to be responsible for paying for the query compute costs.

For information about how to create a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

Topics

- Creating a collaboration
- Creating a membership and joining a collaboration
- Updating a membership
- Editing collaborations
- Change requests in AWS Clean Rooms
- Deleting collaborations
- Viewing collaborations
- Inviting members to a collaboration
- Monitoring members
- Adding members to a collaboration
- Removing members from a collaboration
- Leaving a collaboration

Creating a collaboration

There are three ways to create a collaboration in AWS Clean Rooms.

The most basic form is the collaboration for queries. This collaboration focuses on SQL query analysis and maintains a simple structure with two main roles: one member who can run queries and another who can receive results. This basic collaboration setup works well for simple data analysis tasks.

The second form, collaboration for queries and jobs, extends functionality by incorporating both SQL queries and PySpark jobs. This collaboration setup maintains the same basic role structure but expands permissions to include job execution. A notable requirement is that the member who creates PySpark analysis templates must also be the one receiving results, ensuring clear accountability in the analysis process.

The third form, collaboration for ML modeling, is built for machine learning workflows. This collaboration setup adds two more roles: one for users who need the results from trained models, and another for users who need the results from using those models to make predictions. This collaboration setup helps collaboration members work together on complex data projects while keeping everyone's roles and permissions clear.

The following topics explain how to create collaborations for queries, jobs, and ML modeling.

Topics

- Creating a collaboration for queries
- Creating a collaboration for queries and jobs
- Creating a collaboration for ML modeling

Creating a collaboration for queries

In this procedure, you as the collaboration creator perform the following tasks:

- Create a collaboration.
- Invite one or more members to the collaboration.
- Assign abilities to members, such as the member who can query and the member who can receive results.

If the collaboration creator is also the member who can receive results, they specify the results destination and format. They also provide a service role Amazon Resource Name (ARN) to write the results to the results destination.

- Configure which member is responsible for paying for compute costs in the collaboration.

Before you begin, make sure that you have completed the following prerequisites:

- You have the name and AWS account ID for each member that you want to invite to the collaboration.
- You have permission to share the name and AWS account ID for each member with all members of the collaboration.

> **Note**
>
> You can't add more members after you create the collaboration.

For information about how to create a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

To create a collaboration for queries

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. In the upper right corner, choose **Create collaboration**.
4. For **Step 1: Define collaboration**, do the following:

   a. For **Details**, enter the **Name** and **Description** of the collaboration.

      This information will be visible to collaboration members who are invited to participate in the collaboration. The **Name** and **Description** helps them understand what the collaboration is in reference to.

   b. For **Members**:

      i. For **Member 1: You**, enter your **Member display name** as you want it to appear for the collaboration.

> **Note**
>
> Your AWS account ID is included automatically for **Member AWS account ID**.

ii. For **Member 2**, enter the **Member display name** and **Member AWS account ID** for the member that you want to invite to the collaboration.

The **Member display name** and **Member AWS account ID** will be visible to everyone invited to the collaboration. After you enter and save the values for these fields, you can't edit them.

> **Note**
>
> You must inform the collaboration member that their **Member AWS account ID** and **Member display name** will be visible to all invited and active collaborators in the collaboration.

iii. If you want to add another member, choose **Add another member**. Then enter the **Member display name** and **Member AWS account ID** for each member who can contribute data that you want to invite to the collaboration.

c. If you want to enable **Analysis logging**, select the **Enable analysis logging** checkbox.

- Choose the **Logs from queries** checkbox under the **Supported log types**.

  You will receive logs generated from SQL queries in your Amazon CloudWatch Logs account.

d. If you want to enable **Detailed monitoring**, select the **Enable detailed monitoring** checkbox.

The analysis runner and configured payor can choose to enable detailed metrics when they create their membership. When enabled, detailed monitoring metrics will be published to CloudWatch for operational monitoring of collaborations, including query performance and resource utilization. These metrics will be available to the analysis runner and configured payor in their respective AWS accounts.

For more information about CloudWatch pricing, see CloudWatch Pricing.

e. Under **Allowed query results regions**, select one or more AWS Regions where you want to send query results.

By default, only the current Region (such as N. Virginia us-east-1) is selected.

> **Important**
>
> When you enable cross-Region query results delivery, your results may be processed and stored outside the source Region.

For more information about Regions, see Regions and Endpoints in the *AWS General Reference*.

f. (Optional) Manage access to your data with **Automatic change request approval** by configuring which settings can be changed automatically without manual approvals for a change request. By default, some settings can only be changed by submitting a change request which must be approved by all members before it can take effect.

- **Grant member abilities** – Choose the abilities that can be granted to collaboration members without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results
  - **Auto-approve new members with these abilities** – If allowed, any members added with the abilities selected above will instantly join the collaboration. Members added with other abilities will still require manual approval to join.
- **Abilities that can be automatically revoked** - Choose the abilities that can be revoked without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results

If you choose this option, you can track all collaboration configuration modifications through the **Change requests history**, located on the **Details** tab of the collaboration details page.

g. (Optional) If you want to enable the **Cryptographic computing** capability, select the **Enable cryptographic computing** checkbox.

   i. Choose the following **Cryptographic coverage parameters**:

      - **Allow plaintext columns**

        Choose **No** if you require fully encrypted tables.

        Choose **Yes** if you want cleartext columns allowed in the encrypted table.

        To run SUM or AVG on certain columns, the columns must be in cleartext.

      - **Preserve NULL values**

        Choose **No** if you don't want to preserve NULL values. NULL values won't appear as NULL in an encrypted table.

        Choose **Yes** if you want to preserve NULL values. NULL values will appear as NULL in an encrypted table.

   ii. Choose the following **Fingerprinting parameters**:

      - **Allow duplicates**

        Choose **No** if you don't want duplicate entries allowed in a fingerprint column.

        Choose **Yes** if you want duplicate entries allowed in a fingerprint column.

      - **Allow JOIN of columns with different names**

        Choose **No** if you don't want to join fingerprint columns with different names.

        Choose **Yes** if you want to join fingerprint columns with different names.

For more information about **Cryptographic computing parameters**, see Cryptographic computing parameters.

For more information about how to encrypt your data for use in AWS Clean Rooms, see Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

> **Note**
>
> Verify these configurations carefully before completing the next step. After you create the collaboration, you can only edit the collaboration name, description, and whether the logs are stored in Amazon CloudWatch Logs.

h. If you want to enable **Tags** for the collaboration resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

i. Choose **Next**.

5. For **Step 2: Specify member abilities**, for **Analysis using queries and jobs**, under the **Supported analysis types** leave the **Queries** checkbox selected and take the recommended action, based on your goal.

Your goal	Recommended action
Query the data in the collaboration and receive the results	1. Choose yourself as the member who can **Run queries**. 2. Choose yourself as member who can **Receive results from analyses** from the dropdown list.
Query the data in the collaboration and assign a different member to receive results	1. Choose yourself as the member who can **Run queries**. 2. Select the member who can **Receive results from analyses** from the dropdown list.
Receive the results of the query in the collaboration and assign a different member to query the data	1. Select the member who can **Run queries** from the dropdown list. 2. Choose yourself as member who can **Receive results from analyses** from the dropdown list.
Create and manage the collaboration, assign a different member to query the data, and assign a different member to receive results	1. Select the member who can **Run queries** from the dropdown list.

Your goal	Recommended action
	2. Select the member who can **Receive results from analyses** from the dropdown list.

a. If you are using Clean Rooms ML, for **ML modeling using purpose-built workflows**,

   i. (Optional) Select the member who can **Receive output from trained models** from the dropdown list.

   ii. (Optional) Select the member who can **Receive output from model inference** from the dropdown list.

b. View the member abilities under **ID resolution using AWS Entity Resolution**.

c. Choose **Next**.

6. For **Step 3: Configure payment**, for **Analysis using queries**, take one of the following actions based on your goal.

Your goal	Recommended action
Assign the member who can **Run queries** to be the member who pays for the query compute costs	1. For **Analysis using queries**, choose the member who will **Pay for queries** to be the same as the member who can **Run queries**. 2. Choose **Next**.
Assign a different member to pay for the query compute costs	1. For **Analysis using queries**, choose yourself as the member who will **Pay for queries**. 2. Choose **Next**.

For **ML modeling using purpose-built workflows**, the **Creator of the configured lookalike model** is the member who will **Pay for lookalike modeling**.

For **ID resolution with AWS Entity Resolution**, the **Creator of the ID mapping table** is the member who will **Pay for ID mapping table**.

7. For **Step 4: Configure membership**, choose one of the following options:

   Yes, join by creating membership now

   1. For **Results settings defaults**, for **Query results settings**, if you are the member who can **Receive results**,

      a. For the **Results destination in Amazon S3**, enter the Amazon S3 destination or choose **Browse S3** to select an S3 bucket.

      b. For the query **Result format**, choose either **CSV** or **PARQUET**.

      c. (Spark only) For the **Result files**, choose either **Multiple** or **Single**.

      d. (Optional) For **Service access**, if you want to deliver queries that take up to 24 hours to your S3 destination, select the **Add a service role to support queries that take up to 24 hours to complete** checkbox.

         Large queries that take up to 24 hours to complete will be delivered to your S3 destination.

         If you don't select the checkbox, only queries that complete within 12 hours will be delivered to your S3 location.

      e. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

If you choose to ...	Then ...
Create and use a new service role	• AWS Clean Rooms creates a service role with the required policy for this table. • The default **Service role name** is `cleanrooms-result-receiver-<timestamp>` • You must have permissions to create roles and attach policies.

If you choose to ...	Then ...
Use an existing service role	i. Choose an **Existing service role name** from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use. ii. View the service role by choosing the **View in IAM** external link. If there are no existing service roles, the option to **Use an existing service role** is unavailable. By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

> **Note**
>
> - AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
> - If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
> - If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

2. For **Logs settings**, choose one of the following options for **Log storage in Amazon CloudWatch Logs**:

> **Note**
>
> The **Logs settings** section appears if you chose to enable **Query logging**.

a. Choose **Turn on** and the query logs relevant to you will be stored in your Amazon CloudWatch Logs account.

   Each member can receive only logs for queries that they initiated or that contain their data.

   The member who can receive results also receives logs for all queries run in a collaboration, even if their data isn't accessed in a query.

   Under **Supported log types**, the **Query logs** checkbox is turned on by default.

   > **Note**
   >
   > After you turn on **Query logging**, it can take a few minutes for log storage to be set up and start receiving logs in Amazon CloudWatch Logs. During this brief period, the member who can query might run queries that don’t actually send logs.

b. Choose **Turn off** and the query logs relevant to you won't be stored in your Amazon CloudWatch Logs account.

3. If you want to enable **Tags** for the membership resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
4. If you are the member who is paying for **Query compute**, indicate your acceptance by selecting the **I agree to pay for the compute costs in this collaboration** checkbox.

> **Note**
>
> You must select this checkbox to proceed.
> For more information about how pricing is calculated, see Pricing for AWS Clean Rooms.

If you are the member paying for query compute costs but not the member who can query, it is recommended that you use AWS Budgets to configure a budget for AWS Clean Rooms and receive notifications once the maximum budget has been reached. For more information about setting up a budget, see Managing your costs with AWS Budgets in the *AWS Cost Management User Guide*. For more information about setting up notifications, see Creating an Amazon SNS topic for budget notifications in the *AWS Cost Management User Guide*. If the maximum budget has been reached, you can contact the member who can run queries or leave the collaboration. If you leave the collaboration, no more queries will be allowed to run, and therefore you will no longer be billed for query compute costs.

5. Choose **Next**.

Both the collaboration and your membership are created.

Your status in the collaboration is active.

No, I will create a membership later

1. Choose **Next**.

   Only the collaboration is created.

   Your status in the collaboration is inactive.

8. For **Step 5: Review and create**, do the following:

   a. Review the selections that you made for the previous steps and edit if necessary.

   b. Choose one of the options.

If you have chosen to ...	Then choose ...
Create a membership with the collaboration (**Yes, join by creating membership now**)	**Create collaboration and membership**
Create the collaboration, and not to create a membership at this time (**No, I will create a membership later**)	**Create collaboration**

After your collaboration has been created successfully, you can see the collaboration details page under **Collaborations**.

You are now ready to:

- Prepare your data table to be analyzed in AWS Clean Rooms. (Optional if you want to analyze your own event data or if you want to query identity data.)
- Associate the configured table to your collaboration. (Optional if you want to analyze your own event data.)
- Add an analysis rule for the configured table. (Optional if you want to analyze your own event data.)
- Create a membership and join a collaboration. (Optional if you've already created a membership.)
- Invite members to join the collaboration.

Creating a collaboration for queries and jobs

In this procedure, you as the collaboration creator perform the following tasks:

- Create a collaboration.
- Invite one or more members to the collaboration.
- Assign abilities to members, such as the member who can run queries and jobs and the member who can receive results.

  If the collaboration creator is also the member who can receive results, they specify the results destination and format. They also provide a service role Amazon Resource Name (ARN) to write the results to the results destination.
- Configure which member is responsible for paying for query and job compute costs in the collaboration.

Before you begin, make sure that you have completed the following prerequisites:

- You have the name and AWS account ID for each member that you want to invite to the collaboration.
- You have permission to share the name and AWS account ID for each member with all members of the collaboration.

> **Note**
>
> You can't add more members after you create the collaboration.

For information about how to create a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

To create a collaboration for queries and jobs

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. In the upper right corner, choose **Create collaboration**.
4. For **Step 1: Define collaboration**, do the following:
   a. For **Details**, enter the **Name** and **Description** of the collaboration.

      This information will be visible to collaboration members who are invited to participate in the collaboration. The **Name** and **Description** helps them understand what the collaboration is in reference to.

   b. For **Members**:

      i. For **Member 1: You**, enter your **Member display name** as you want it to appear for the collaboration.

      > **Note**
      >
      > Your AWS account ID is included automatically for **Member AWS account ID**.

      ii. For **Member 2**, enter the **Member display name** and **Member AWS account ID** for the member that you want to invite to the collaboration.

      The **Member display name** and **Member AWS account ID** will be visible to everyone invited to the collaboration. After you enter and save the values for these fields, you can't edit them.

> **Note**
>
> You must inform the collaboration member that their **Member AWS account ID** and **Member display name** will be visible to all invited and active collaborators in the collaboration.

iii. If you want to add another member, choose **Add another member**. Then enter the **Member display name** and **Member AWS account ID** for each member who can contribute data that you want to invite to the collaboration.

c. If you want to enable **Analysis logging**, select the **Enable analysis logging** checkbox, and then choose the **Supported log types**.

- If you want to receive logs generated from SQL queries, choose the **Logs from queries** checkbox.
- If you want to receive logs generated from jobs using PySpark, choose the **Logs from jobs** checkbox.

d. If you want to enable **Detailed monitoring**, select the **Enable detailed monitoring** checkbox.

The analysis runner and configured payor can choose to enable detailed metrics when they create their membership. When enabled, detailed monitoring metrics will be published to CloudWatch for operational monitoring of collaborations, including query performance and resource utilization. These metrics will be available to the analysis runner and configured payor in their respective AWS accounts.

For more information about CloudWatch pricing, see CloudWatch Pricing.

e. Under **Allowed query results regions**, select one or more AWS Regions where you want to send query results.

By default, only the current Region (such as N. Virginia us-east-1) is selected.

> **Important**
>
> When you enable cross-Region query results delivery, your results may be processed and stored outside the source Region.

For more information about Regions, see Regions and Endpoints in the *AWS General Reference*.

f. (Optional) Manage access to your data with **Automatic change request approval** by configuring which settings can be changed automatically without manual approvals for a change request. By default, some settings can only be changed by submitting a change request which must be approved by all members before it can take effect.

- **Grant member abilities** – Choose the abilities that can be granted to collaboration members without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results
  - **Auto-approve new members with these abilities** – If allowed, any members added with the abilities selected above will instantly join the collaboration. Members added with other abilities will still require manual approval to join.
- **Abilities that can be automatically revoked** - Choose the abilities that can be revoked without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results

If you choose this option, you can track all collaboration configuration modifications through the **Change requests history**, located on the **Details** tab of the collaboration details page.

g. (Optional) If you want to enable the **Cryptographic computing** capability, select the **Enable cryptographic computing** checkbox.

   i. Choose the following **Cryptographic coverage parameters**:

      - **Allow plaintext columns**

        Choose **No** if you require fully encrypted tables.

        Choose **Yes** if you want cleartext columns allowed in the encrypted table.

To run SUM or AVG on certain columns, the columns must be in cleartext.

- **Preserve NULL values**

  Choose **No** if you don't want to preserve NULL values. NULL values won't appear as NULL in an encrypted table.

  Choose **Yes** if you want to preserve NULL values. NULL values will appear as NULL in an encrypted table.

ii. Choose the following **Fingerprinting parameters**:

- **Allow duplicates**

  Choose **No** if you don't want duplicate entries allowed in a fingerprint column.

  Choose **Yes** if you want duplicate entries allowed in a fingerprint column.

- **Allow JOIN of columns with different names**

  Choose **No** if you don't want to join fingerprint columns with different names.

  Choose **Yes** if you want to join fingerprint columns with different names.

For more information about **Cryptographic computing parameters**, see Cryptographic computing parameters.

For more information about how to encrypt your data for use in AWS Clean Rooms, see Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

> **Note**
>
> Verify these configurations carefully before completing the next step. After you create the collaboration, you can only edit the collaboration name, description, and whether the logs are stored in Amazon CloudWatch Logs.

h. If you want to enable **Tags** for the collaboration resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

i. Choose **Next**.

5. For **Step 2: Specify member abilities**, do the following:

a. For **Analysis using queries and jobs**, under **Supported analysis types**, choose the **Jobs** checkbox.

   The **Queries** checkbox is selected by default.

   i. Select the member who can **Run queries and jobs** from the dropdown list.

   ii. Select the member who can **Receive results from analyses** from the dropdown list.

   > **Note**
   >
   > The member who creates the PySpark analysis template must also be the member who receives results.

b. If you are using Clean Rooms ML, for **ML modeling using purpose-built workflows**,

   i. (Optional) Select the member who can **Receive output from trained models** from the dropdown list.

   ii. (Optional) Select the member who can **Receive output from model inference** from the dropdown list.

c. View the member abilities under **ID resolution using AWS Entity Resolution**.

d. Choose **Next**.

6. For **Step 3: Configure payment**,

   a. For **Analysis using queries and jobs**, choose the member who will **Pay for queries and jobs**.

      You can assign the member who can **Run queries and jobs** to be the member who pays for the queries and jobs compute costs.

      You can assign a different member to pay for the queries and jobs compute costs.

   b. For **ML modeling using purpose-built workflows**, the **Creator of the configured lookalike model** is the member who will **Pay for lookalike modeling**.

   c. For **ID resolution with AWS Entity Resolution**, the **Creator of the ID mapping table** is the member who will **Pay for ID mapping table**.

   d. Choose **Next**.

7. For **Step 4: Configure membership**, choose one of the following options:

Yes, join by creating membership now

1. For **Results settings defaults**, for **Query results settings**, if you are the member who can **Receive results**,
   a. Choose the **Set default settings for queries** checkbox. For the **Results destination in Amazon S3**, enter the Amazon S3 destination or choose **Browse S3** to select an S3 bucket.
   b. For the query **Result format**, choose either **CSV** or **PARQUET**.
   c. (Spark only) For the **Result files**, choose either **Multiple** or **Single**.
   d. (Optional) For **Service access**, if you want to deliver queries that take up to 24 hours to your S3 destination, select the **Add a service role to support queries that take up to 24 hours to complete** checkbox.

      Large queries that take up to 24 hours to complete will be delivered to your S3 destination.

      If you don't select the checkbox, only queries that complete within 12 hours will be delivered to your S3 location.
   e. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

If you choose to ...	Then ...
Create and use a new service role	• AWS Clean Rooms creates a service role with the required policy for this table. • The default **Service role name** is `cleanrooms-result-receiver-<timestamp>` • You must have permissions to create roles and attach policies.

If you choose to ...	Then ...
Use an existing service role	i. Choose an **Existing service role name** from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use. ii. View the service role by choosing the **View in IAM** external link. If there are no existing service roles, the option to **Use an existing service role** is unavailable. By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

Note

- AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
- If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
- If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

2. For **Job results**,

Example

For example: `s3://bucket/prefix`

a. Choose the **Set default settings for jobs** checkbox, and then specify the **Results destination in Amazon S3** by entering the S3 destination or choose **Browse S3** to select from a list of available S3 buckets.

b. Specify the **Service access** permissions by choosing an **Existing service role name** from the dropdown list.

3. For **Logs settings**, choose one of the following options for **Log storage in Amazon CloudWatch Logs**:

> **Note**
>
> The **Logs settings** section appears if you chose to enable **Query logging**.

a. Choose **Turn on** and the query logs relevant to you will be stored in your Amazon CloudWatch Logs account.

   Each member can receive only logs for queries that they initiated or that contain their data.

   The member who can receive results also receives logs for all queries run in a collaboration, even if their data isn't accessed in a query.

   Under **Supported log types**, choose from the log types the collaboration creator has chosen to support:

   Under **Supported log types**, the **Query logs** and **Job logs** checkboxes are turned on by default.

> **Note**
>
> After you turn on **Analysis logging**, it can take a few minutes for log storage to be set up and start receiving logs in Amazon CloudWatch Logs. During this

brief period, the member who can query might run queries that don't actually send logs.

b. Choose **Turn off** and the query logs relevant to you won't be stored in your Amazon CloudWatch Logs account.

4. If you want to enable **Membership tags** for the membership resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
5. If you are the member who is paying for **Query compute**, or **Job compute**, or both, indicate your acceptance by selecting the **I agree to pay for the compute costs in this collaboration** checkbox.

> **Note**
>
> You must select this checkbox to proceed.
> For more information about how pricing is calculated, see Pricing for AWS Clean Rooms.

If you are the member paying for query compute costs but not the member who can query, it is recommended that you use AWS Budgets to configure a budget for AWS Clean Rooms and receive notifications once the maximum budget has been reached. For more information about setting up a budget, see Managing your costs with AWS Budgets in the *AWS Cost Management User Guide*. For more information about setting up notifications, see Creating an Amazon SNS topic for budget notifications in the *AWS Cost Management User Guide*. If the maximum budget has been reached, you can contact the member who can run queries or leave the collaboration. If you leave the collaboration, no more queries will be allowed to run, and therefore you will no longer be billed for query compute costs.

6. Choose **Next**.

Both the collaboration and your membership are created.

Your status in the collaboration is active.

No, I will create a membership later

1. Choose **Next**.

Only the collaboration is created.

Your status in the collaboration is inactive.

8. For **Step 5: Review and create**, do the following:

   a. Review the selections that you made for the previous steps and edit if necessary.

   b. Choose one of the options.

If you have chosen to ...	Then choose ...
Create a membership with the collaboration (**Yes, join by creating membership now**)	**Create collaboration and membership**
Create the collaboration, and not to create a membership at this time (**No, I will create a membership later**)	**Create collaboration**

After your collaboration has been created successfully, you can see the collaboration details page under **Collaborations**.

You are now ready to:

- Prepare your data table to be analyzed in AWS Clean Rooms. (Optional if you want to analyze your own event data or if you want to query identity data.)
- Associate the configured table to your collaboration. (Optional if you want to analyze your own event data.)
- Add an analysis rule for the configured table. (Optional if you want to analyze your own event data.)
- Create a membership and join a collaboration. (Optional if you've already created a membership.)
- Invite members to join the collaboration.

Creating a collaboration for ML modeling

In this procedure, you as the collaboration creator perform the following tasks:

- Create a collaboration.
- Invite one or more members to the collaboration.
- Assign abilities to members, such as the
  - Member who can query
  - Member who can receive results
  - Member who can receive output from trained models
  - Member who can output from model inference

  If the collaboration creator is also the member who can receive results, they specify the results destination and format. They also provide a service role Amazon Resource Name (ARN) to write the results to the results destination.
- Configure which member is responsible for paying for compute costs, model training, and model inference costs in the collaboration.

Before you begin, make sure that you have completed the following prerequisites:

- You have the name and AWS account ID for each member that you want to invite to the collaboration.
- You have permission to share the name and AWS account ID for each member with all members of the collaboration.

> **Note**
>
> You can't add more members after you create the collaboration.

For information about how to create a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

To create a collaboration for ML modeling

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. In the upper right corner, choose **Create collaboration**.
4. For **Step 1: Define collaboration**, do the following:

a. For **Details**, enter the **Name** and **Description** of the collaboration.

   This information will be visible to collaboration members who are invited to participate in the collaboration. The **Name** and **Description** helps them understand what the collaboration is in reference to.

b. For **Members**:

   i. For **Member 1: You**, enter your **Member display name** as you want it to appear for the collaboration.

   > **Note**
   >
   > Your AWS account ID is included automatically for **Member AWS account ID**.

   ii. For **Member 2**, enter the **Member display name** and **Member AWS account ID** for the member that you want to invite to the collaboration.

   The **Member display name** and **Member AWS account ID** will be visible to everyone invited to the collaboration. After you enter and save the values for these fields, you can't edit them.

   > **Note**
   >
   > You must inform the collaboration member that their **Member AWS account ID** and **Member display name** will be visible to all invited and active collaborators in the collaboration.

   iii. If you want to add another member, choose **Add another member**. Then enter the **Member display name** and **Member AWS account ID** for each member who can contribute data that you want to invite to the collaboration.

c. If you want to enable **Analysis logging**, select the **Enable analysis logging** checkbox, and then under **Supported log types**, choose **Logs from queries**.

d. If you want to enable **Detailed monitoring**, select the **Enable detailed monitoring** checkbox.

   The analysis runner and configured payor can choose to enable detailed metrics when they create their membership. When enabled, detailed monitoring metrics will be

published to CloudWatch for operational monitoring of collaborations, including query performance and resource utilization. These metrics will be available to the analysis runner and configured payor in their respective AWS accounts.

For more information about CloudWatch pricing, see CloudWatch Pricing.

e. Under **Allowed query results regions**, select one or more AWS Regions where you want to send query results.

By default, only the current Region (such as N. Virginia us-east-1) is selected.

> **Important**
>
> When you enable cross-Region query results delivery, your results may be processed and stored outside the source Region.

For more information about Regions, see Regions and Endpoints in the *AWS General Reference*.

f. (Optional) Manage access to your data with **Automatic change request approval** by configuring which settings can be changed automatically without manual approvals for a change request. By default, some settings can only be changed by submitting a change request which must be approved by all members before it can take effect.

- **Grant member abilities** – Choose the abilities that can be granted to collaboration members without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results
  - **Auto-approve new members with these abilities** – If allowed, any members added with the abilities selected above will instantly join the collaboration. Members added with other abilities will still require manual approval to join.
- **Abilities that can be automatically revoked** - Choose the abilities that can be revoked without manual approval. Members can always contribute data.
  - Choose abilities:
    - Contribute data *(always enabled)*
    - Receive results

If you choose this option, you can track all collaboration configuration modifications through the **Change requests history**, located on the **Details** tab of the collaboration details page.

g. (Optional) If you want to enable the **Cryptographic computing** capability, select the **Enable cryptographic computing** checkbox.

   i. Choose the following **Cryptographic coverage parameters**:

      - **Allow plaintext columns**

        Choose **No** if you require fully encrypted tables.

        Choose **Yes** if you want cleartext columns allowed in the encrypted table.

        To run SUM or AVG on certain columns, the columns must be in cleartext.

      - **Preserve NULL values**

        Choose **No** if you don't want to preserve NULL values. NULL values won't appear as NULL in an encrypted table.

        Choose **Yes** if you want to preserve NULL values. NULL values will appear as NULL in an encrypted table.

   ii. Choose the following **Fingerprinting parameters**:

      - **Allow duplicates**

        Choose **No** if you don't want duplicate entries allowed in a fingerprint column.

        Choose **Yes** if you want duplicate entries allowed in a fingerprint column.

      - **Allow JOIN of columns with different names**

        Choose **No** if you don't want to join fingerprint columns with different names.

        Choose **Yes** if you want to join fingerprint columns with different names.

For more information about **Cryptographic computing parameters**, see Cryptographic computing parameters.

For more information about how to encrypt your data for use in AWS Clean Rooms, see Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

> **Note**
>
> Verify these configurations carefully before completing the next step. After you create the collaboration, you can only edit the collaboration name, description, and whether the logs are stored in Amazon CloudWatch Logs.

h. If you want to enable **Tags** for the collaboration resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

i. Choose **Next**.

5. For **Step 2: Specify member abilities**,

   a. For **Analysis using queries and jobs**, under the **Supported analysis types**, leave the **Queries** checkbox selected.

   b. For **Run queries**, choose the member who will initiate the model training

   c. For **Receive results from analyses**, choose one or more members who will receive the query results.

   d. For **ML modeling using purpose-built workflows**,

      i. For **Receive output from trained models**, choose the member who will receive trained model results, including model artifacts and metrics.

      ii. For **Receive output from model inference**, choose the member who will receive the model inference results.

   e. View the member abilities under **ID resolution using AWS Entity Resolution**.

6. For **Step 3: Configure payment**,

   a. Under **Analysis using queries**, for **Pay for queries**, do one of the following actions:

      - To have the same member pay for and run queries, select the same member you chose for **Run queries**.
      - To have a different member pay for query costs, select your member account.

   b. For **ML modeling using purpose-built workflows**,

      - Choose the member who will **Pay for model training**.

c. Choose the member who will **Pay for inference job**.

d. For **Pay for lookalike modeling**, no action is needed. The **Creator of the configured lookalike model** is the member who will pay for lookalike modeling.

e. (Optional) Choose the member who will **Pay for Synthetic data generation**.

f. For **ID resolution with AWS Entity Resolution**, no action is needed. The **Creator of the ID mapping table** is the member who will **Pay for ID mapping table**.

7. Choose **Next**.

8. For **Step 4: Configure membership**, under **Collaboration membership**, choose one of the following options:

   Yes, join by creating membership now

   1. For **Results settings defaults**, for **Query results settings**, if you are the member who can **Receive results**,

      a. Select the **Set default settings for queries** checkbox.

      b. For the **Results destination in Amazon S3**, enter the Amazon S3 destination or choose **Browse S3** to select an S3 bucket.

      c. For the query **Result format**, choose either **CSV** or **PARQUET**.

      d. (Spark only) For the **Result files**, choose either **Multiple** or **Single**.

      e. (Optional) If you want to deliver queries that take up to 24 hours to your S3 destination, select the **Add a service role to support queries that take up to 24 hours to complete** checkbox.

         Large queries that take up to 24 hours to complete will be delivered to your S3 destination.

         If you don't select the checkbox, only queries that complete within 12 hours will be delivered to your S3 location.

      f. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

If you choose to ...	Then ...
Create and use a new service role	• AWS Clean Rooms creates a service role with the required policy for this table. • The default **Service role name** is `cleanrooms-result-receiver-<timestamp>` • You must have permissions to create roles and attach policies.
Use an existing service role	i. Choose an **Existing service role name** from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use. ii. View the service role by choosing the **View in IAM** external link. If there are no existing service roles, the option to **Use an existing service role** is unavailable. By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

> **Note**
>
> - AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
> - If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
> - If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

2. For **ML configuration**,
   a. Choose the **Create ML configuration** checkbox, and then specify the **Model output destination on Amazon S3** by entering the S3 destination or choose **Browse S3** to select from a list of available S3 buckets.
   b. Specify the **Service access** permissions by choosing to either **Create and use a new service role** or **Use an existing service role**.
   c. If the S3 bucket is encrypted, select the **The destination bucket is encrypted with a KMS key** checkbox and then enter the **AWS KMS key** or select **Create an AWS KMS key** to create a new KMS key.
3. If you want to enable **Membership tags** for the membership resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
4. If you are the member who is paying for **Query compute**, indicate your acceptance by selecting the **I agree to pay for the compute costs in this collaboration** checkbox.

> **Note**
>
> You must select this checkbox to proceed.
> For more information about how pricing is calculated, see Pricing for AWS Clean Rooms.

If you are the member paying for query compute costs but not the member who can query, it is recommended that you use AWS Budgets to configure a budget for AWS Clean Rooms and receive notifications once the maximum budget has been reached. For more information about setting up a budget, see Managing your costs with AWS Budgets in the *AWS Cost Management User Guide*. For more information about setting up notifications, see Creating an Amazon SNS topic for budget notifications in the *AWS Cost Management User Guide*. If the maximum budget has been reached, you can contact the member who can run queries or leave the collaboration. If you leave the collaboration, no more queries will be allowed to run, and therefore you will no longer be billed for query compute costs.

5. Choose **Next**.

Both the collaboration and your membership are created.

Your status in the collaboration is active.

No, I will create a membership later

1. Choose **Next**.

   Only the collaboration is created.

   Your status in the collaboration is inactive.

9. For **Step 5: Review and create**, do the following:

   a. Review the selections that you made for the previous steps and edit if necessary.

   b. Choose one of the options.

If you have chosen to ...	Then choose ...
Create a membership with the collaboration (**Yes, join by creating membership now**)	**Create collaboration and membership**
Create the collaboration, and not to create a membership at this time (**No, I will create a membership later**)	**Create collaboration**

Creating a membership and joining a collaboration

A *membership* is a resource that's created when a member joins a collaboration in AWS Clean Rooms.

You can join a collaboration as a

- member who can query
- member who can run queries and jobs
- member who can receive results of a query or a job
- member paying for query compute costs
- member paying for queries and jobs

All members can contribute data.

For information about how to create a membership and join a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

In this procedure, the invited member joins the collaboration by creating a membership resource.

If the invited member is the member who can receive results, they specify the results destination and format. They also provide a service role ARN to write to the results destination.

If the invited member is the member who is responsible to pay for compute costs, they accept their payment responsibilities before joining the collaboration.

To create a membership and join a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Available to join** tab, for **Collaborations available to join**, choose the **Name** of the collaboration.
4. On the collaboration details page, in the **Overview** section, view the collaboration details, including **Your member details** and a list of the other members.

   Verify that the AWS account IDs for each member of the collaboration are the ones with whom you intend to enter in to the collaboration.

5. Choose **Create membership**.
6. On the **Create membership** page, in the **Overview**, view the **Collaboration name**, **Collaboration description**, AWS account ID of the **Collaboration creator**, **Your member details**, and the AWS account ID of the member who will **Pay for queries**.
7. If the collaboration creator has chosen to enable **Analysis logging**, choose one of the following options for **Log storage in Amazon CloudWatch Logs**:

If you choose...	Then ...
Turn on	The logs relevant to you are stored in Amazon CloudWatch Logs. Each member can receive only logs for queries that they initiated or that contain their data. The member who can receive results also receives logs for all analyses run in a collaboration, even if their data isn't accessed in an analysis. Under **Supported log types**, choose from the log types the collaboration creator has chosen to support: 1. If you want to receive logs generated from SQL queries, choose the **Logs from queries** checkbox. 2. If you want to receive logs generated from jobs using PySpark, choose the **Logs from jobs** checkbox.
Turn off	The query logs relevant to you aren't stored in your Amazon CloudWatch Logs account.

> **Note**
>
> After you turn on **Analysis logging**, it can take a few minutes for log storage to be set up and start receiving logs in Amazon CloudWatch Logs. During this brief period, the member who can query might run queries that don't actually send logs.

8. If the collaboration creator enabled **Detailed monitoring** for this collaboration, choose whether you want to receive detailed observability metrics in your CloudWatch account.

   For **Detailed monitoring**:

   Detailed monitoring options

Option	Description
Turn on	AWS Clean Rooms will publish detailed monitoring metrics to CloudWatch for this collaboration in your account. You can use these metrics for operational monitoring, including query performance and resource utilization. Additional CloudWatch charges apply. For more information, see CloudWatch Pricing.
Turn off	No detailed metrics will be exported to your CloudWatch account. Other members can still view detailed monitoring metrics in their own accounts if they have enabled this option.

> **Note**
>
> Detailed monitoring metrics are only available to members who can run queries (analysis runners) and members who are configured as payors for the collaboration.

9. If **Your member abilities** includes **Receive results**, for **Results settings defaults**:

   a. For **Query results**, choose the **Set default settings for queries** checkbox, and then specify the **Results destination in Amazon S3** by entering the S3 destination or choose **Browse S3** to select from a list of available S3 buckets.

Example

For example: **`s3://bucket/prefix`**

i. For the **Result format**, choose either **CSV** or **PARQUET**.

ii. (Spark only) For the **Result files**, choose either **Multiple** or **Single**.

iii. (Optional) For **Service access**, if you want to deliver queries that take up to 24 hours to your S3 destination, select the **Add a service role to support queries that take up to 24 hours to complete** check box.

Large queries that take up to 24 hours to complete will be delivered to your S3 destination.

If you don't select the check box, only queries that complete within 12 hours will be delivered to your S3 location.

> **Note**
>
> You must either select an existing service role or have permissions to create a new one. For more information, see Create a service role to receive results.

iv. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

Create and use a new service role

- AWS Clean Rooms creates a service role with the required policy for this table.
- The default **Service role name** is `cleanrooms-result-receiver-<timestamp>`
- You must have permissions to create roles and attach policies.

Use an existing service role

1. Choose an **Existing service role name** from the dropdown list.

   The list of roles are displayed if you have permissions to list roles.

If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.

2. View the service role by choosing the **View in IAM** external link.

   If there are no existing service roles, the option to **Use an existing service role** is unavailable.

   By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

> **Note**
>
> - AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
> - If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
> - If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

b. For **Job results**, choose the **Set default settings for jobs** checkbox, and then specify the **Results destination in Amazon S3** by entering the S3 destination or choose **Browse S3** to select from a list of available S3 buckets.

   Example

   For example: `s3://bucket/prefix`

   - Specify the **Service access** permissions by choosing an **Existing service role name** from the dropdown list.

10. If you want to enable **Tags** for the membership resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

11. If the collaboration creator has designated you as the member who will **Pay for queries** or **Pay for queries and jobs**, indicate your acceptance by selecting the **I agree to pay for the compute costs in this collaboration** checkbox.

> **Note**
>
> You must select this checkbox to proceed.
> For more information about how pricing is calculated, see Pricing for AWS Clean Rooms.

If you are the member paying for query compute costs or the member paying for queries and job compute costs but not the member who can query, it is recommended that you use AWS Budgets to configure a budget for AWS Clean Rooms and receive notifications once the maximum budget has been reached. For more information about setting up a budget, see Managing your costs with AWS Budgets in the *AWS Cost Management User Guide*. For more information about setting up notifications, see Creating an Amazon SNS topic for budget notifications in the *AWS Cost Management User Guide*. If the maximum budget has been reached, you can contact the member who can run queries and jobs or leave the collaboration. If you leave the collaboration, no more queries will be allowed to run, and therefore you will no longer be billed for query compute costs.

12. If you are sure that you want to create a membership and join the collaboration, choose **Create membership**.

You are given read access to the collaboration metadata. This includes information such as the display name and description of the collaboration, in addition to all the names and AWS account IDs of other members.

You are now ready to:

- Prepare your data table to be queried in AWS Clean Rooms. (Optional if you want to query your own event data or if you want to query identity data.)
- Associate the configured table to your collaboration – if you want to query event data.
- Add an analysis rule for the configured table – if you want to query event data.
- Create and associate a new ID namespace – if you want to create an ID mapping table to query identity data.

For information about how to leave a collaboration, see Leaving a collaboration.

Updating a membership

After you join a collaboration, you can update your membership settings. You can change your analysis logging settings, default result settings, and payment configuration.

For information about how to update a membership using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

You must already be a member of a collaboration (have an active membership) to update membership settings. The available settings depend on your member abilities in the collaboration.

Topics

- Update analysis logging settings
- Update default result settings
- Update payment configuration

Update analysis logging settings

You can update your analysis logging settings to turn on or turn off log storage in Amazon CloudWatch Logs.

To update analysis logging settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Memberships**.
3. Choose the membership name.
4. On the membership detail page, choose the **Logging** tab.
5. Choose **Edit logging settings**.
6. For **Log storage in Amazon CloudWatch Logs**, choose **Turn on** or **Turn off**.
7. Choose **Save changes**.

> **Note**
>
> After you turn on or turn off **Analysis logging**, it can take a few minutes for the changes to take effect.

Update default result settings

You can update the default results destination in Amazon S3, the result format, and the service role for your membership.

To update default result settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Memberships**.
3. Choose the membership name.
4. On the membership detail page, choose the **Results settings** tab.
5. Choose **Edit result settings**.
6. Update the result settings as needed:

   a. For **Results destination in Amazon S3**, enter the S3 destination or choose **Browse S3** to select from a list of available S3 buckets.

      Example

      For example: **`s3://bucket/prefix`**

   b. For **Result format**, choose either **CSV** or **PARQUET**.

   c. For **Service access**, choose either **Create and use a new service role** or **Use an existing service role**.

      Create and use a new service role

      - AWS Clean Rooms creates a service role with the required policy.
      - The default **Service role name** is `cleanrooms-result-receiver-<timestamp>`
      - You must have permissions to create roles and attach policies.

Use an existing service role

1. Choose an **Existing service role name** from the dropdown list.

   If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.

> **Note**
>
> For more information about the required service role permissions, see Create a service role to receive results.

7. Choose **Save changes**.

Update payment configuration

You can update your payment responsibilities for the membership.

To update payment configuration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Memberships**.
3. Choose the membership name.
4. On the membership detail page, choose the **Payment configuration** tab.
5. Choose **Edit payment configuration**.
6. Update your payment responsibilities as needed:

   a. If you are the member paying for query compute costs, select or clear the **I agree to pay for the query compute costs in this collaboration** checkbox to accept or decline payment responsibility.

   b. If you are the member paying for query and job compute costs, select or clear the **I agree to pay for the compute costs in this collaboration** checkbox to accept or decline payment responsibility for both query and job compute costs.

7. Choose **Save changes**.

> **Note**
>
> We recommend that you use AWS Budgets to configure a budget for AWS Clean Rooms and receive notifications when the maximum budget is reached. For more information about pricing, see Pricing for AWS Clean Rooms.

For more information about related tasks, see the following topics:

- Creating a membership and joining a collaboration
- Editing collaborations

Editing collaborations

As a collaboration creator, you can edit the different parts of a collaboration.

For information about how to edit a collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

Topics

- Edit collaboration name and description
- Update collaboration analytics engine
- Turn off log storage
- Edit collaboration logs settings
- Edit collaboration tags
- Edit membership tags
- Add a new member
- Edit existing member abilities
- Edit collaboration auto-approval settings
- Edit associated table tags
- Edit analysis template tags
- Edit differential privacy policy tags

Edit collaboration name and description

After you create the collaboration, you can only edit the collaboration name and description.

To edit the collaboration name and description

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. On the collaboration detail page, choose **Actions**, and then choose **Edit collaboration**.
5. On the **Edit collaboration** page, for **Details**, edit the **Name** and **Description** of the collaboration.
6. Choose **Save changes**.

Update collaboration analytics engine

After you create the collaboration, you can change the analytics engine from AWS Clean Rooms SQL to Spark.

> **Note**
>
> Changing the analytics engine from AWS Clean Rooms SQL to Spark might break existing workflows.

To update the collaboration analytics engine

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. On the collaboration detail page, choose **Actions**, and then choose **Edit collaboration**.
5. Choose **Save changes**.

Turn off log storage

If you have enabled **Analysis logging**, you can edit whether the analysis logs are stored in your Amazon CloudWatch Logs account.

To turn off log storage

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has analysis logging turned on.
4. On the collaboration detail page, choose **Actions**, and then choose **Turn off log storage**.

> **Note**
>
> A warning appears, indicating the following:
>
> - New queries will no longer be logged to your CloudWatch account.
> - Existing logs will be preserved according to your current retention settings.
> - If you reactivate logging in the future, it will only apply to queries made after reactivation.
> - This change affects only your logs - other team members' logging settings remain unchanged.

5. Choose **Turn off**.

Edit collaboration logs settings

If you have enabled **Query logging**, you can edit whether the query logs are stored in your Amazon CloudWatch Logs account.

To edit collaboration logs settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.

4. On the collaboration detail page, do one of the following:
   - Choose **Actions**, and then choose **Edit logs settings**.
   - On the **Logs** tab, choose **Edit logs settings**.
5. On the **Edit logs settings modal**, for **Log storage in Amazon CloudWatch Logs**:
   - If you don't want logs relevant to you to be stored in your Amazon CloudWatch Logs account, choose **Turn off**.
   - If you do want logs relevant to you to be stored in your Amazon CloudWatch Logs account, choose **Turn on**.

     You can only receive logs for queries that you initiated or that contain you data.

     The member who can receive results also receives logs for all queries run in a collaboration, even if their data isn't accessed in a query.

     1. Under **Supported log types**, choose from the log types the collaboration creator has chosen to support:
        - If you want to receive logs generated from SQL queries, choose the **Logs from queries** checkbox.
        - If you want to receive logs generated from jobs using PySpark, choose the **Logs from jobs** checkbox.
6. Choose **Save changes**.

> **Note**
>
> After you turn on logging, it can take a few minutes for log storage to be set up and start receiving logs in Amazon CloudWatch Logs. During this brief period, the member who can query might run queries that don’t actually send logs.

Edit collaboration tags

As a collaboration creator, after you have created a collaboration, you can manage the tags on the collaboration resource.

To edit the collaboration tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose one of the following:

If you are...	Then ...
The collaboration creator and a member of the collaboration	Choose the **Details** tab.
The collaboration creator but not a member of the collaboration	Scroll down the page to the **Tags** section.

5. For **Collaboration details**, choose **Manage tags**.
6. On the **Manage tags** page, you can do the following:
   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.
   - To save your changes, choose **Save changes**

Edit membership tags

As a collaboration creator, after you have created a collaboration, you can manage the tags on the membership resource.

To edit the membership tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose the **Details** tab.
5. For **Membership details**, choose **Manage tags**.

6. On the **Manage membership tags** page, you can do the following:

   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.
   - To save your changes, choose **Save changes**.

Add a new member

For more information, see Add a new member to a collaboration.

Edit existing member abilities

For more information, see Update member abilities.

Edit collaboration auto-approval settings

For more information, see Edit collaboration auto-approval settings.

Edit associated table tags

As a collaboration creator, after you associate tables to a collaboration, you can manage the tags on the associated table resource.

To edit the associated table tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose the **Tables** tab.
5. For **Tables associated by you**, choose a table.
6. On the configured table detail page, for **Tags**, choose **Manage tags**.

   On the **Manage tags** page, you can do the following:

   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.

- To save your changes, choose **Save changes**.

Edit analysis template tags

As a collaboration creator, after you have created a collaboration, you can manage the tags on the analysis template resource.

To edit the membership tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose the **Templates** tab.
5. On the **Analysis templates created by you** section, choose the analysis template.
6. On the analysis template table detail page, scroll down to the **Tags** section.
7. Choose **Manage tags**.
8. On the **Manage tags** page, you can do the following:
   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.
   - To save your changes, choose **Save changes**.

Edit differential privacy policy tags

As a collaboration creator, after you have created a collaboration, you can manage the tags on the analysis template resource.

To edit the membership tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that contains the differential privacy policy you want to edit.
4. Choose the **Tables** tab.

5. On the **Tables** tab, choose the **Manage tags**.
6. On the **Manage tags** page, you can do the following:
   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.
   - To save your changes, choose **Save changes**.

Change requests in AWS Clean Rooms

Change requests allow you to propose changes to existing collaboration settings for approval by other collaboration members. With change requests, you can submit a request to add new members, update member abilities, update ML member abilities, update payer configurations, and modify collaboration auto-approval settings. All collaboration members must approve change requests for the proposed changes to take effect.

Change requests are collaboration-specific and can be submitted by the collaboration creator.

You can submit a change request in the following ways:

- Add a new member to a collaboration
- Update member abilities
- Update ML member abilities
- Update payer configurations
- Edit collaboration auto-approval settings

> **Note**
>
> You must be the collaboration creator to submit a change request.

Add a new member to a collaboration

To add a new member to a collaboration, you must be the collaboration creator. Adding a new member to a collaboration requires manual approval from existing collaboration members, and will submit a change request.

You can add new members to a collaboration by following these steps:

To add a new member to a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. Select your collaboration to navigate to your collaboration details.
4. On the collaboration page, navigate to the **Members** tab.
5. In the Members table, select **Edit members**.
6. Choose **Add another member**.
7. Input the new member information:
   - Member display name
   - Member AWS account ID
   - Specify whether the member can receive results. Check the box to grant this member ability.
8. Choose **Save changes**.
9. Confirm your change request submission. In the confirmation modal, confirm the changes and select **Submit change request**.

> **Note**
>
> If auto-approved change types are supported in your collaboration, manual approval of change requests may not be required. You can review which change types do not require a change request in the "Overview" section of the collaboration. For more information, see the section called “Edit collaboration auto-approval settings”.

Update member abilities

To update existing collaboration member abilities, you must be the collaboration creator. Updating abilities for existing collaboration members requires manual approval from existing members, and will submit a change request.

The member abilities that can be updated are:

- Can receive results

- Export query analysis log

For more information about exporting query analysis logs, see Exporting query analysis logs.

> **Note**
>
> When you update member abilities through the API, the abilities that you specify replace all existing abilities. To keep an existing ability, include it along with the ability that you're adding.

> **Note**
>
> To update ML member abilities, see the section called "Update ML member abilities".

You can update the member abilities for existing collaboration members by following these steps:

To update member abilities

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. Select your collaboration to navigate to your collaboration details.
4. On the collaboration page, navigate to the **Members** tab.
5. In the Members table, select **Edit members**.
6. Specify the member ability to change.
7. Choose **Save changes**.
8. Confirm your change request submission. In the confirmation modal, confirm the changes and select **Submit change request**.

> **Note**
>
> If auto-approved change types are supported in your collaboration, manual approval of change requests may not be required. You can review which change types do not require a

change request in the "Overview" section of the collaboration. For more information, see the section called “Edit collaboration auto-approval settings”.

Update ML member abilities

To update ML member abilities for existing collaboration members, you must be the collaboration creator. Updating ML member abilities requires manual approval from existing members, and will submit a change request.

The ML member abilities that can be updated are:

- Can receive model output
- Can receive inference output

You can update the ML member abilities for existing collaboration members by following these steps:

To update ML member abilities

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. Select your collaboration to navigate to your collaboration details.
4. On the collaboration page, navigate to the **Members** tab.
5. In the Members table, select **Edit members**.
6. Specify the ML member abilities to change for the member.
7. Choose **Save changes**.
8. Confirm your change request submission. In the confirmation modal, confirm the changes and select **Submit change request**.

> **Note**
>
> If auto-approved change types are supported in your collaboration, manual approval of change requests may not be required. You can review which change types do not require a

change request in the "Overview" section of the collaboration. For more information, see the section called “Edit collaboration auto-approval settings”.

Update payer configurations

To update payer configurations for a collaboration, you must be the collaboration creator. Updating payer configurations requires manual approval from existing members, and will submit a change request.

The payer configurations that can be updated are:

- Query compute
- Job compute
- Machine learning (model training, model inference, and synthetic data generation)

You can update the payer configurations for a collaboration by following these steps:

To update payer configurations

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. Select your collaboration to navigate to your collaboration details.
4. On the collaboration page, navigate to the **Payment configuration** tab.
5. Choose **Edit payment configuration**.
6. Update the payer configurations for the collaboration members.
7. Choose **Save changes**.
8. Confirm your change request submission. In the confirmation modal, confirm the changes and select **Submit change request**.

> **Note**
>
> If auto-approved change types are supported in your collaboration, manual approval of change requests may not be required. You can review which change types do not require a

change request in the "Overview" section of the collaboration. For more information, see the section called “Edit collaboration auto-approval settings”.

Edit collaboration auto-approval settings

To edit the collaboration settings for auto-approvals, you must be the collaboration creator and submit a change request for approval by other collaboration members.

You can edit the auto-approval settings for a collaboration by following these steps:

To edit collaboration auto-approval settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your member AWS account.
2. In the left navigation pane, choose **Collaborations**.
3. Select your collaboration to navigate to your collaboration details.
4. On the collaboration page, choose the **Actions** button and select **Edit auto approvals**.
5. **To grant member abilities to existing members without manually approving a change request:**

   1. Navigate to the **Grant member abilities** section.
   2. Specify which member abilities can be automatically granted.

      > **Note**
      >
      > By default, all collaboration members can "Contribute data".

      You can grant the export query analysis logs ability automatically, without a separate approval cycle.
   3. (Optional) To allow new members to instantly join a collaboration with the specified abilities without manually approving change requests, select **Auto-approve new members with these abilities**.
6. **To allow removal of abilities from existing members without a change request:**

   1. Navigate to the **Abilities that can be automatically revoked** section.

   2. Specify which member abilities can be automatically removed.
7. Choose **Save changes**.
8. Confirm your change request submission. In the confirmation modal, confirm the changes and select **Submit change request**.

Deleting collaborations

As a collaboration creator, you can delete a collaboration that you created.

> **Note**
>
> When you delete a collaboration, you and all members can't run queries, receive results, or contribute data. Each collaboration member continues to have access to their own data as part of their membership.

To delete a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you want to delete.
4. Under **Actions**, choose **Delete collaboration**.
5. Confirm the deletion and then choose **Delete**.

Viewing collaborations

As a collaboration creator, you can view all of the collaborations that you created.

To view collaborations

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, under **Last used**, view the last 5 collaborations used.
4. On the **With active membership** tab, view the list of **Collaborations with active membership**.

   You can sort by **Name**, the **Membership created date**, and **Your member details**.

You can use the **Search** bar to search for a collaboration.

5. On the **Available to join** tab, view the list of **Collaborations available to join**.
6. On the **No longer available** tab, view the list of deleted collaborations and **Memberships for collaborations that are no longer available** (removed memberships).

Inviting members to a collaboration

As a collaboration creator, after you have created a collaboration, you can send an invitation link to the members listed on the **Members** tab.

To invite members to a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose the **Members** tab.
5. In the **Members** table, choose the **Copy invitation link** button.

   The invitation link is copied.
6. Paste the invitation link in the secure communication method of your choice and send it to each member of the collaboration.

Monitoring members

As a collaboration creator, after you have created a collaboration, you can monitor the status of all members on the **Members** tab. This helps ensure proper access control and cost management across your collaboration.

To monitor the status of a member

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that you created.
4. Choose the **Members** tab.
5. In the **Members** table, review the **Status** of each member.

6. In the **Member abilities** table, review which members can query, receive results, contribute data, and perform other tasks.
7. In the **Payment configuration** table, review which members are paying for queries, ID mapping tables, and ML modeling.

Adding members to a collaboration

Prerequisites

- An AWS account with permissions to manage collaborations
- The AWS account IDs of members you want to add

To add a member to a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Select the collaboration you want to add members to.
4. Choose the **Members** tab.
5. Choose **Edit members**.
6. Choose **Add another member** and enter the following information:
   - **Member display name**
   - **Member AWS account ID**
   - Specify whether they **can receive results**
7. Choose **Save changes**.
8. In the confirmation modal, verify that the information is correct, and choose **Submit change request**.
   - If the change request requires other members' approval, all existing members must approve the change request before the new member is added. For more information on change requests, see the section called “Change requests”.
   - If auto-approval change types are supported for new members with the specified abilities, then this change will take effect immediately. For more information, see the section called “Edit collaboration auto-approval settings”.

9. On the collaboration detail page, under the **Members** tab, verify that the **Member status** of the added member(s) displays as **Invited**.

After completing these steps, the invited members can join the collaboration. For more information about joining a collaboration, see Creating a membership and joining a collaboration

Removing members from a collaboration

> **Note**
>
> Before you begin, note that removing a member:
>
> - Removes all of their associated datasets from the collaboration
> - If the member pays for query compute costs, this action stops all query execution in the collaboration.
>
> Removing a member also removes all of their associated datasets from the collaboration.

Prerequisites

- You must be a collaboration creator
- You can't remove your own account

To remove a member from a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Select the collaboration you want to modify.
4. Choose the **Members** tab.
5. Select the option button next to the member you want to remove.
6. Choose **Remove**.
7. In the confirmation dialog box, type **`confirm`** to verify the removal.

> **Note**
>
> After you remove a member, all datasets associated with their account are also removed from the collaboration.

> **Important**
>
> If you remove the member who pays for query compute costs, no further queries can run in the collaboration until you designate a new paying member.

Leaving a collaboration

As a collaboration member, you can leave a collaboration by deleting your membership. If you are the collaboration creator, you can only leave a collaboration by deleting the collaboration.

> **Note**
>
> When you delete your membership, you leave the collaboration and can't re-join it. If you are the member paying for query compute costs and you delete your membership, no more queries are allowed to run.

To leave a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. For **With active membership**, choose the collaboration of which you are a member.
4. Choose **Actions**.
5. Choose **Delete membership**.
6. In the dialog box, confirm the decision to leave the collaboration by typing **`confirm`** in the text input field, and then choose **Empty and delete membership**.

   You see a message on the console indicating that the membership was deleted.

   The collaboration creator sees the **Member status** as **Left**.

Prepare data tables in AWS Clean Rooms

> **Note**
>
> Preparing data tables can take place before or after you have joined a collaboration. After a table is prepared, you can reuse it across multiple collaborations as long as your privacy needs for that table are the same.

As a member in the collaboration, you must prepare your data tables before they can be queried in AWS Clean Rooms by the collaboration member who can query.

The data tables that you use for queries in AWS Clean Rooms are commonly the same types of data tables that you use for other applications. For example, the same types of datasets are used with Amazon Athena, Amazon EMR, Amazon Redshift Spectrum, and Amazon Quick.

You can query the data in its original format directly from any of the following data sources:

- Amazon Simple Storage Service (Amazon S3)
- Amazon Athena
- Snowflake

AWS Clean Rooms accesses the dataset at query run time, ensuring that members who can query always access the most up-to-date data. Any data that is temporarily read into an AWS Clean Rooms collaboration is deleted after the query completes. The query results are written to your Amazon S3 bucket.

If your use case involves querying identity data, see AWS Entity Resolution in AWS Clean Rooms.

Topics

- Data formats for AWS Clean Rooms
- Apache Iceberg tables in AWS Clean Rooms
- Preparing data tables for queries in AWS Clean Rooms
- Preparing encrypted data tables with Cryptographic Computing for Clean Rooms
- Decrypting data tables with the C3R encryption client

Data formats for AWS Clean Rooms

To analyze data, the datasets must be in a format that AWS Clean Rooms supports.

Topics

- Supported data formats for PySpark jobs
- Supported data formats for SQL queries
- Supported data types
- File compression types for AWS Clean Rooms
- Server-side encryption for AWS Clean Rooms

Supported data formats for PySpark jobs

AWS Clean Rooms supports the following structured formats for running PySpark jobs.

- Parquet
- OpenCSV
- JSON

Supported data formats for SQL queries

AWS Clean Rooms supports the following structured formats for running SQL queries.

- Apache Iceberg tables
- Parquet
- OpenCSV
- JSON

> **Note**
>
> A `timestamp` value in a text file must be in the format `yyyy-MM-dd HH:mm:ss.SSSSSS`. For example: `2017-05-01 11:30:59.000000`.

We recommend using a columnar storage file format, such as Apache Parquet. With a columnar storage file format, you can minimize data movement by selecting only the columns that you need. For optimal performance, large objects should be split into 100mb–1gb objects.

Supported data types

AWS Clean Rooms supports the following data types.

- ARRAY
- BIGINT
- BOOLEAN
- BYTE
- CHAR
- DATE
- DECIMAL
- FLOAT
- INTEGER
- INTERVAL
- LONG
- MAP
- REAL
- SHORT
- SMALLINT
- STRUCT
- TIME
- TIMESTAMP_LTZ
- TIMESTAMP_NTZ
- TINYINT
- VARCHAR

For more information, see Data types in the *AWS Clean Rooms SQL Reference*.

File compression types for AWS Clean Rooms

To reduce storage space, improve performance, and minimize costs, we strongly recommend that you compress your datasets.

AWS Clean Rooms recognizes file compression types based on the file extension and supports the compression types and extensions shown in the following table.

Compression algorithm	File extension
GZIP	.gz
Bzip2	.bz2
Snappy	.snappy
Zstandard (Zstd)	.zstd

You can apply compression at different levels. Most commonly, you compress a whole file or compress individual blocks within a file. Compressing columnar formats at the file level doesn't yield performance benefits.

Server-side encryption for AWS Clean Rooms

> **Note**
>
> Server-side encryption does not replace cryptographic computing for those use cases that require it.

AWS Clean Rooms transparently decrypts datasets that are encrypted using the following encryption options:

- **SSE-S3** – Server-side encryption using an AES-256 encryption key managed by Amazon S3
- **SSE-KMS** – Server-side encryption with keys managed by AWS Key Management Service

To use SSE-S3, the AWS Clean Rooms service role used to associate the configured table to the collaboration must have KMS-decrypt permissions. To use SSE-KMS, the KMS key policy must also allow the AWS Clean Rooms service role to decrypt.

AWS Clean Rooms doesn't support Amazon S3 client-side encryption. For more information about server-side encryption, see Protecting data using server-side encryption in the *Amazon Simple Storage Service User Guide*.

Apache Iceberg tables in AWS Clean Rooms

Apache Iceberg is an open source table format for data lakes. AWS Clean Rooms can use the statistics stored in Apache Iceberg metadata to optimize query plans and reduce file scans during clean room query processing. For more information, see the Apache Iceberg documentation.

Consider the following when using AWS Clean Rooms with Iceberg tables:

- **Apache Iceberg tables for S3** – Apache Iceberg tables must be defined in the AWS Glue Data Catalog based on the open source glue catalog implementation.
- **Apache Iceberg tables for Athena** – For more information, see https://docs.aws.amazon.com/athena/latest/ug/querying-iceberg.html
- **Apache Iceberg tables for Snowflake** – For more information, see https://docs.snowflake.com/en/user-guide/tables-iceberg
- **Parquet file format** – AWS Clean Rooms only supports Iceberg tables in the Parquet data file format.
- **GZIP, Snappy, and Zstd compression** – AWS Clean Rooms supports Parquet with GZIP, Snappy, and Zstandard (Zstd) compression.
- **Iceberg versions** – AWS Clean Rooms supports running queries against version 1 and version 2 Iceberg tables.
- **Partitions** – You don't need to manually add partitions for your Apache Iceberg tables in AWS Glue. AWS Clean Rooms detects new partitions in Apache Iceberg tables automatically and no manual operation is needed to update partitions in the table definition. Iceberg partitions appear as regular columns in the AWS Clean Rooms table schema and not separately as a partition key in the configured table schema.
- **Limitations**
  - **New Iceberg tables only**

Apache Iceberg tables converted from Apache Parquet tables are not supported.

- **Time travel queries**

  AWS Clean Rooms does not support time travel queries with Apache Iceberg tables.

- **Athena engine version 2**

  Iceberg tables created with Athena engine version 2 are not supported.

- **File formats**

  Avro and Optimized Row Columnar (ORC) file formats are not supported.

Supported data types for Iceberg tables

AWS Clean Rooms can query Iceberg tables that contain the following data types:

- BOOLEAN
- DATE
- DECIMAL
- DOUBLE
- FLOAT
- INT
- LIST
- LONG
- MAP
- STRING
- STRUCT
- TIMESTAMP WITHOUT TIME ZONE

For more information about Iceberg data types, see the Schemas for Iceberg in the Apache Iceberg documentation.

Preparing data tables for queries in AWS Clean Rooms

If your use case doesn't require you to bring your own data, you can skip this procedure.

If your use case involves querying identity data, see AWS Entity Resolution in AWS Clean Rooms.

For more information about the data formats that you can use, see Data formats for AWS Clean Rooms.

Topics

- Preparing data tables in Amazon S3
- Preparing data tables in Amazon Athena
- Preparing data tables in Snowflake

Preparing data tables in Amazon S3

You can analyze data tables that have been cataloged in AWS Glue and stored in Amazon S3. If your data tables are already cataloged in AWS Glue, skip to Creating a configured table in AWS Clean Rooms.

> **Important**
>
> When preparing your data in Amazon S3 for use with AWS Clean Rooms, you must ensure that your AWS Glue table location exactly matches the directory path where your data files are stored.
> For example: If your data is stored in `s3://mybucket/folder/subfolder/data.parquet`, your AWS Glue table location must point to `s3://mybucket/folder/subfolder/`. Setting the table location to a parent directory (`s3://mybucket/folder/`) will result in the table appearing empty when queried.

Preparing your data tables in Amazon S3 involves the following steps:

Topics

- Step 1: Complete the prerequisites
- Step 2: (Optional) Prepare your data for cryptographic computing
- Step 3: Upload your data table to Amazon S3
- Step 4: Create an AWS Glue table
- Step 5: Next steps

Step 1: Complete the prerequisites

To prepare your data tables for use with AWS Clean Rooms, you must complete the following prerequisites:

- Your data tables are saved as one of the supported data formats for AWS Clean Rooms.
- Your data tables are cataloged in AWS Glue and use the supported data types for AWS Clean Rooms.
- All of your data tables are stored in Amazon Simple Storage Service (Amazon S3).
- The AWS Glue Data Catalog is in the same AWS account as the membership.
- The Amazon S3 bucket isn't registered with AWS Lake Formation.

Step 2: (Optional) Prepare your data for cryptographic computing

(Optional) If you're using cryptographic computing and your data table contains sensitive information that you want to encrypt, you must encrypt the data table using the C3R encryption client.

To prepare your data for cryptographic computing, follow the procedures in Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

Step 3: Upload your data table to Amazon S3

> **Note**
>
> If you intend to use encrypted data tables in the collaboration, you must first encrypt the data for cryptographic computing before you upload your data table to Amazon S3. For more information, see Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

To upload your data table to Amazon S3

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose **Buckets**, and then choose a bucket where you want to store your data table.
3. Choose **Upload**, and then follow the prompts.

4. Choose the **Objects** tab to view the prefix where your data is stored. Make a note of the name of the folder.

   You can select the folder to view the data.

Step 4: Create an AWS Glue table

If you already have an AWS Glue data table, you can skip this step.

In this step, you set up a crawler in AWS Glue that crawls all the files in your S3 bucket and creates an AWS Glue table. For more information, see Defining crawlers in AWS Glue in the *AWS Glue User Guide*.

For more information about supported AWS Glue Data Catalog data types, see Supported data types.

> **Note**
>
> AWS Clean Rooms doesn't currently support S3 buckets registered with AWS Lake Formation.

The following procedure describes how to create an AWS Glue table. If you want to use an encrypted AWS Glue Data Catalog object with an AWS Key Management Service (AWS KMS) key, you need to configure the KMS key permissions policy to allow access to that encrypted table. For more information, see Setting up encryption in AWS Glue in the *AWS Glue Developer Guide*.

To create an AWS Glue table

1. Follow the Working with crawlers on the AWS Glue console procedure in the *AWS Glue User Guide*.
2. Make a note of the AWS Glue database name and AWS Glue table name.

Step 5: Next steps

Now that you have prepared your data tables in Amazon S3, you are ready to:

- Create a configured table
- Create an ML model

The tables can be queried after:

- The collaboration creator has set up a collaboration in AWS Clean Rooms. For more information, see Creating a collaboration.
- The collaboration creator has sent the collaboration ID to you as a participant in the collaboration.

Preparing data tables in Amazon Athena

You can query data tables that have been created as AWS Glue Data Catalog (GDC) Views in Amazon Athena.

A GDC View is a virtual table, created from one or more underlying AWS Glue tables. It must be created using Athena SQL in the Athena `AwsGlueCatalog` catalog.

> **Note**
>
> Complex (nested) data types—ARRAY, MAP, and STRUCT—are not supported for Amazon Athena data sources in AWS Clean Rooms. If an underlying AWS Glue table or GDC View contains a column of a complex data type, queries fail during table setup. This occurs even when the query doesn't reference that column. To work around this limitation, create a GDC View that excludes the complex-typed columns, or cast them to a supported type such as STRING.

Preparing your data tables in Amazon Athena involves the following steps:

Topics

- Step 1: Complete the prerequisites
- Step 2: Next steps

Step 1: Complete the prerequisites

To prepare your data tables for use with AWS Clean Rooms, you must complete the following prerequisites:

- Your data tables are saved as one of the supported data formats for AWS Clean Rooms.

- Your data tables use the supported data types for AWS Clean Rooms.
- You have created a GDC View on your AWS Glue table using Athena SQL in the Athena `AwsDataCatalog` catalog.

  The view will appear in:

  - The Athena console (under the `AwsDataCatalog`) as a View: https://console.aws.amazon.com/athena/
  - The AWS Glue console as a AWS Glue table: https://console.aws.amazon.com/glue/

  For more information, see Use Data Catalog views in Athena in the *Amazon Athena User Guide*.

  > **Note**
  >
  > You need appropriate permissions to create Views in Athena and AWS Glue. Also, ensure that you have access to the underlying tables referenced in your View definition.
  > AWS Clean Rooms only supports the AWS Glue Catalog Type for Athena, not Lambda or Hive Catalog Types.

- Your data tables or GDC Views are cataloged in AWS Glue and are registered with AWS Lake Formation.
- You have created a separate output bucket in Amazon S3 to receive the Athena results.
- You have set up a service role to read the data from Amazon Athena. For more information, see Create a service role to read data from Amazon Athena.
  - The service role has Lake Formation Select and Describe access permissions on the GDC View or table.

Step 2: Next steps

Now that you have prepared your data tables in Amazon Athena, you are ready to:

- Create a configured table
- Create an ML model

The tables can be queried after:

- The collaboration creator has set up a collaboration in AWS Clean Rooms. For more information, see Creating a collaboration.
- The collaboration creator has sent the collaboration ID to you as a participant in the collaboration.

Preparing data tables in Snowflake

You can query data tables that have been stored in Snowflake data warehouse.

Preparing your data tables in Snowflake involves the following steps:

Topics

- Step 1: Complete the prerequisites
- Step 2: (Optional) Prepare your data for cryptographic computing
- Step 3: Create an AWS Secrets Manager secret
- Step 4: Next steps

Step 1: Complete the prerequisites

To prepare your data tables for use with AWS Clean Rooms, you must complete the following prerequisites:

- You have an AWS account with the proper permissions granted to read your data tables. For more information, see Create a service role to read data from Snowflake.
- Your data tables are saved as one of the supported data formats for AWS Clean Rooms.
- Your data tables use the supported data types for AWS Clean Rooms.
- Your data table is stored in a Snowflake warehouse. For more information, see the Snowflake documentation.
- You have set up a new Snowflake user with read-only privileges to the Snowflake table you are going to associate with your collaboration.

Step 2: (Optional) Prepare your data for cryptographic computing

(Optional) If you're using cryptographic computing and your data table contains sensitive information that you want to encrypt, you must encrypt the data table using the C3R encryption client.

To prepare your data for cryptographic computing, follow the procedures in Preparing encrypted data tables with Cryptographic Computing for Clean Rooms.

Step 3: Create an AWS Secrets Manager secret

To connect to Snowflake from AWS Clean Rooms, you will need to create and store your Snowflake credentials in a AWS Secrets Manager secret, then associate that secret with a Snowflake table in AWS Clean Rooms.

> **Note**
>
> We recommend that you create a new user that is exclusively for AWS Clean Rooms. That user should only have a role with Read permissions for the data that you want AWS Clean Rooms to access.

To create an AWS Secrets Manager secret

1. In Snowflake, generate a user `snowflakeUser` and set up key-pair authentication.

   > **Note**
   >
   > In November 2025, Snowflake will transition to supporting only key-pair authentication. This change will affect the current AWS Clean Rooms integration with Snowflake, which uses username and password authentication. After this date, Snowflake connections in AWS Clean Rooms will require key-pair authentication using a Snowflake Privacy Enhanced Mail (PEM) private key.

2. Determine which Snowflake warehouse this user will interact with, `snowflakeWarehouse`. Either set it as the `DEFAULT_WAREHOUSE` for `snowflakeUser` in Snowflake or remember it for the next step.
3. In AWS Secrets Manager, create a secret using your Snowflake credentials. To create a secret in Secrets Manager, follow the tutorial available in Create an AWS Secrets Manager secret in the *AWS Secrets Manager User Guide*. After creating the secret, keep the Secret name, `secretName` for the next step.

   - When selecting **Key/value pairs**, create a pair for `snowflakeUser` with the key `sfUser`.

- When selecting **Key/value pairs**, create a pair for your Snowflake PEM private key with the key `pem_private_key`.
- When selecting **Key/value pairs**, create a pair for `snowflakeWarehouse` with the key `sfWarehouse`.

  This isn't needed if a default is set in Snowflake.
- When selecting **Key/value pairs**, create a pair for `snowflakeRole` with the key `sfRole`.

Step 4: Next steps

Now that you have prepared your data tables in Snowflake, you are ready to:

- Create a configured table
- Create an ML model

The tables can be queried after:

- The collaboration creator has set up a collaboration in AWS Clean Rooms. For more information, see Creating a collaboration.
- The collaboration creator has sent the collaboration ID to you as a participant in the collaboration.

Preparing encrypted data tables with Cryptographic Computing for Clean Rooms

Cryptographic Computing for Clean Rooms (C3R) is a capability in AWS Clean Rooms. You can use C3R to limit cryptographically what can be learned by any party and AWS in an AWS Clean Rooms collaboration.

You can encrypt the data table using the C3R encryption client, a client-side encryption tool, before uploading the data table to your data source: Amazon Simple Storage Service (Amazon S3), Amazon Athena, or Snowflake.

For more information, see Cryptographic Computing for Clean Rooms.

Preparing encrypted data tables with C3R involves the following steps:

Steps

- Step 1: Complete the prerequisites
- Step 2: Download the C3R encryption client
- Step 3: (Optional) View available commands in the C3R encryption client
- Step 4: Generate an encryption schema for a tabular file
- Step 5: Create a shared secret key
- Step 6: Store the shared secret key in an environment variable
- Step 7: Encrypt data
- Step 8: Verify data encryption
- (Optional) Create a schema (advanced users)

Step 1: Complete the prerequisites

To prepare your data tables for use with C3R, you must complete the following prerequisites:

- You can access the Cryptographic Computing for Clean Rooms repository on GitHub:

  https://github.com/aws/c3r
- You have set up AWS credentials to use the C3R encryption client. These credentials are used by C3R encryption client for read-only API calls to AWS Clean Rooms to retrieve collaboration metadata. For more information, see Configuring the AWS CLI in the *AWS Command Line Interface User Guide for Version 2*.
- You have Java Runtime Environment (JRE) 11 or later installed on your machine.
  - The recommended Java Runtime Environment, Amazon Corretto 11 or higher, can be downloaded from https://aws.amazon.com/corretto.
  - The Java Development Kit (JDK) includes a corresponding JRE of the same version. However, the additional capabilities of the JDK are not needed for running the Cryptographic Computing for Clean Rooms (C3R) encryption client.
- Your tabular data files (.csv) or Parquet files (.parquet) are saved locally.
- You or another member in the collaboration has the ability to create a shared secret key. For more information, see Step 5: Create a shared secret key.
- The collaboration creator has created a collaboration in AWS Clean Rooms with **Cryptographic computing** enabled for the collaboration. For more information, see Creating a collaboration.

- The collaboration creator has sent the collaboration ID to you as a participant in the collaboration. The collaboration Amazon Resource Name (ARN) is included in the invitation that is sent, which contains the collaboration ID.

Step 2: Download the C3R encryption client

To download the C3R encryption client from GitHub

1. Go to the Cryptographic Computing for Clean Rooms AWS GitHub repository: https://github.com/aws/c3r
2. Select and download the files.

   The source code, licenses, and related material can be cloned or downloaded as a .zip file from the GitHub repository's landing page. (See the **Code** button at the top-right of the repository's content list).

   The latest signed C3R encryption client Java Executable File (that is, the command line interface application) is on the **Releases** page of the GitHub repository.

   The C3R encryption client package for Apache Spark (`c3r-cli-spark`) is a version of the c3r-cli that must be submitted as a job to a running Apache Spark server. For more information, see Running C3R on Apache Spark.

Step 3: (Optional) View available commands in the C3R encryption client

Use this procedure to familiarize yourself with the available commands in the C3R encryption client.

To view all of the available commands in the C3R encryption client

1. From a command line interface (CLI), navigate to the folder that contains the downloaded c3r-cli.jar file.
2. Run the following command: `java -jar c3r-cli.jar`
3. View the list of available commands and options.

Step 4: Generate an encryption schema for a tabular file

To encrypt data, an encryption schema describing how the data will be used is required. This section describes how the C3R encryption client assists in generating an encryption schema for a CSV file with a header row or a Parquet file.

You only need to do this once per file. After the schema exists, it can be re-used to encrypt the same file (or any file with identical column names). If the column names or desired encryption schema changes, you must update the schema file. For more information, see (Optional) Create a schema (advanced users).

> **Important**
>
> It is paramount that all collaborating parties use the same shared secret key. Collaborating parties should also coordinate column names to match if they will be JOINed or otherwise compared for equality in queries. Otherwise, the SQL queries might produce unexpected or incorrect results. However, this is not necessary if the collaboration creator enabled the `allowJoinsOnColumnsWithDifferentNames` encryption setting during collaboration creation. For more information about encryption-relevant settings, see Cryptographic computing parameters.

When run in schema mode, the C3R encryption client goes through the input file column by column, prompting you if and how that column should be treated. If the file contains many columns that aren't wanted for the encrypted output, the interactive schema generation might become tedious because you must skip each undesired column. To avoid this, you could manually write a schema, or create a simplified version of the input file featuring only the wanted columns. Then, the interactive schema generator could be run on that reduced file. The C3R encryption client outputs information about the schema file and asks you how the source columns should be included or encrypted (if at all) in the target output.

For each source column in the input file, you are prompted for:

1. How many target columns should be generated
2. How each target column should be encrypted (if at all)
3. The name of each target column
4. How data should be padded before encryption if the column is being encrypted as a sealed column

> **Note**
>
> When you encrypt data for a column that has been encrypted as a sealed column, you must determine which data needs padding. The C3R encryption client suggests a default padding during schema generation that pads all entries in a column to the same length.
> When determining the length for `fixed`, note that padding is in bytes, not bits.

The following is a decision table for creating the schema.

Schema decision table

Decision	Number of target columns from source column <'name-of-column'> ?	Target column type: [c] cleartext , [f] fingerprint, or [s] sealed ?	Target column headernam e <default 'name-of-column'>	Add suffix <suffix> to header to indicate how it was encrypted, [y] yes or [n] no <default 'yes'>	<'name-of-column_sealed'> padding type: [n] one, [f] fixed, or [m] max <default 'max'>
Leave the column unencrypted.	1	**c**	Not applicabl e	Not applicabl e	Not applicabl e
Encrypt the column as a fingerprint column.	1	**f**	Choose default or enter a new header name.	Enter y to choose default (`_fingerpr int`) or enter n.	Not applicabl e
Encrypt the column as a sealed column.	1	**s**	Choose default or enter a new header name.	Enter y to choose default	Choose padding type .

Decision	Number of target columns from source column <‘name-of-column’> ?	Target column type: [c] cleartext , [f] fingerprint, or [s] sealed ?	Target column headername <default 'name-of-column'>	Add suffix <suffix> to header to indicate how it was encrypted, [y] yes or [n] no <default 'yes'>	<‘name-of-column_sealed’> padding type: [n] one, [f] fixed, or [m] max <default ’max’>
				(`_sealed`) or enter n.	For more information, see (Optional) Create a schema (advanced users).
Encrypt the column as both fingerprint and sealed.	2	Enter first target column: **f** . Enter second target column: **s**.	Choose the target headers for each target column.	Enter y to choose default or enter n **.**	Choose padding type (for sealed columns only). For more information, see (Optional) Create a schema (advanced users).

The following are two examples of how to create encryption schemas. The exact content of your interaction depends on the input file and the responses that you provide.

Examples

- Example: Generate an encryption schema for a fingerprint column and a cleartext column
- Example: Generate an encryption schema with sealed, fingerprint, and cleartext columns

Example: Generate an encryption schema for a fingerprint column and a cleartext column

In this example, for `ads.csv`, there are only two columns: `username` and `ad_variant`. For these columns, we want the following:

- For the `username` column to be encrypted as a `fingerprint` column
- For the `ad_variant` column to be a `cleartext` column

To generate an encryption schema for a fingerprint column and a cleartext column

1. (*Optional*) To ensure the c3r-cli.jar file and file to be encrypted are present:

   a. Navigate to the desired directory and run `ls` (if using a Mac or Unix/Linux) or `dir` if using Windows).

   b. View the list of tabular data files (for example, .csv) and choose a file to encrypt.

      In this example, `ads.csv` is the file that we want to encrypt.

2. From the CLI, run the following command to create a schema interactively.

```
java -jar c3r-cli.jar schema ads.csv --interactive --output=ads.json
```

> **Note**
>
> - You can run `java --jar PATH/TO/c3r-cli.jar`. Or, if you have added `PATH/TO/c3r-cli.jar` to your CLASSPATH environment variable, you can also run the class name. The C3R encryption client will look in the CLASSPATH to find it (for example, `java com.amazon.psion.cli.Main`).
> - The `--interactive` flag selects the interactive mode for developing the schema. This walks the user through a wizard for creating the schema. Users with advanced skills can create their own schema JSON without using the wizard. For more information, see (Optional) Create a schema (advanced users).

- The `--output` flag sets an output name. If you don't include the `--output` flag, the C3R encryption client tries to pick a default output name (such as `<input>.out.csv` or for the schema, `<input>.json`).

3. For `Number of target columns from source column ‘username’?`, enter **1** and then press **Enter**.
4. For `Target column type: [c]leartext, [f]ingerprint, or [s]ealed?`, enter **f** and then press **Enter**.
5. For `Target column headername <default 'username'>`, press **Enter**.

   The default name ‘`username`’ is used.
6. For `Add suffix '_fingerprint' to header to indicate how it was encrypted, [y]es or [n]o <default 'yes'>`, enter **y** and then press **Enter**.

   > **Note**
   >
   > The interactive mode suggests suffixes to add to the encrypted column headers (`_fingerprint` for fingerprint columns and `_sealed` for sealed columns). The suffixes might be helpful when you're performing tasks such as uploading data to AWS services or creating AWS Clean Rooms collaborations. These suffixes can help indicate what can be done with the encrypted data in each column. For example, things will not work if you encrypt a column as a sealed column (`_sealed`) and try to JOIN on it or try the reverse.

7. For `Number of target columns from source column ‘ad_variant’?`, enter **1** and then press **Enter**.
8. For `Target column type: [c]leartext, [f]ingerprint, or [s]ealed?`, enter **c** and then press **Enter**.
9. For `Target column headername <default 'username'>`, press **Enter**.

   The default name ‘`ad_variant`’ is used.

   The schema is written to a new file called `ads.json`.

> **Note**
>
> You can view the schema by opening it in any text editor, such as Notepad on Windows or TextEdit on macOS.

10. You are now ready to encrypt data.

Example: Generate an encryption schema with sealed, fingerprint, and cleartext columns

In this example, for `sales.csv`, there are three columns: `username` , `purchased`, and `product`. For these columns, we want the following:

- For the `product` column to be a `sealed` column
- For the `username` column to be encrypted as a `fingerprint` column
- For the `purchased` column to be a `cleartext` column

To generate an encryption schema with sealed, fingerprint, and cleartext columns

1. (*Optional*) To ensure the c3r-cli.jar file and file to be encrypted are present:

   a. Navigate to the desired directory and run `ls` (if using a Mac or Unix/Linux) or `dir` if using Windows).

   b. View the list of tabular data files (.csv) and choose a file to encrypt.

      In this example, `sales.csv` is the file that we want to encrypt.

2. From the CLI, run the following command to create a schema interactively.

```
java -jar c3r-cli.jar schema sales.csv --interactive --
output=sales.json
```

> **Note**
>
> - The `--interactive` flag selects the interactive mode for developing the schema. This walks the user through a guided workflow for creating the schema.

- If you are an advanced user, you can create your own schema JSON without using the guided workflow. For more information, see (Optional) Create a schema (advanced users).
- For .csv files with no column headers, see the `--noHeaders` flag for the schema command available in the CLI.
- The `--output` flag sets an output name. If you don't include the `--output` flag, the C3R encryption client tries to pick a default output name (such as `<input>.out` or for the schema, `<input>.json`).

3. For `Number of target columns from source column ‘username’?`, enter **1** and then press **Enter**.
4. For `Target column type: [c]leartext, [f]ingerprint, or [s]ealed?`, enter **f** and then press **Enter**.
5. For `Target column headername <default 'username'>`, press **Enter**.

   The default name ‘`username`’ is used.
6. For `Add suffix '_fingerprint' to header to indicate how it was encrypted, [y]es or [n]o <default 'yes'>`, enter **y** and then press **Enter**.
7. For `Number of target columns from source column ‘purchased’?`, enter **1** and then press **Enter**.
8. For `Target column type: [c]leartext, [f]ingerprint, or [s]ealed?`, enter **c** and then press **Enter**.
9. For `Target column headername <default 'purchased'>`, press **Enter**.

   The default name ‘`purchased`’ is used.
10. For `Number of target columns from source column ‘product’?`, enter **1** and then press **Enter**.
11. For `Target column type: [c]leartext, [f]ingerprint, or [s]ealed?`, enter **s** and then press **Enter**.
12. For `Target column headername <default 'product'>`, press **Enter**.

    The default name ‘`product`’ is used.
13. For `‘product_sealed’ padding type: [n]one, [f]ixed, or [m]ax <default ’max’?>`, press **Enter** to choose the default.

14. For `Byte-length beyond max length to pad cleartext to in 'product_sealed' <default '0'>?` press **Enter** to choose the default.

    The schema is written to a new file called `sales.json`.

15. You are now ready to encrypt data.

Step 5: Create a shared secret key

To encrypt the data tables, the collaboration participants must agree upon and securely share a shared secret key.

The shared secret key must be at least 256-bits (32 bytes). You can specify a larger key, but it won't give you any additional security.

> **Important**
>
> Remember, the key and collaboration ID used for encryption and decryption must be identical for all collaboration participants.

The following sections provide examples of console commands for generating a shared secret key saved as `secret.key` in the respective terminal's current working directory.

Topics

- Example: Key generation using OpenSSL
- Example: Key generation on Windows using PowerShell

Example: Key generation using OpenSSL

For a common general purpose cryptography library, run the following command to create a shared secret key.

```
openssl rand 32 > secret.key
```

If you're using Windows and don't have OpenSSL installed, you can generate keys using the example described in Example: Key generation on Windows using PowerShell.

Example: Key generation on Windows using PowerShell

For PowerShell, a terminal application available on Windows, run the following command to create a shared secret key.

```
$bs = New-Object Byte[](32);
[Security.Cryptography.RandomNumberGenerator]::Create().GetBytes($bs); Set-
Content 'secret.key' -Encoding Byte -Value $bs
```

Step 6: Store the shared secret key in an environment variable

An environment variable is a convenient and extensible way for users to provide a secret key from various key stores like AWS Secrets Manager and pass it to the C3R encryption client.

The C3R encryption client can use keys stored in AWS services if you use the AWS CLI to store those keys in the relevant environment variable. For example, the C3R encryption client can use a key from AWS Secrets Manager. For more information, see Create and manage secrets with AWS Secrets Manager in the *AWS Secrets Manager User Guide*.

> **Note**
>
> However, before you use an AWS service such as AWS Secrets Manager to hold your C3R keys, verify that your use case permits it. Certain use cases might require that the key be withheld from AWS. This is to ensure that the encrypted data and the key are never held by the same third party.

The only requirements for a shared secret key are that the shared secret key is base64-encoded and stored in the environment variable `C3R_SHARED_SECRET`.

The following sections describe the console commands for converting a `secret.key` file to base64 and storing it as an environment variable. The `secret.key` file could have been generated from any of the commands listed in Step 5: Create a shared secret key and is only an example source.

Store key in an environment variable on Windows using PowerShell

To convert to base64 and set the environment variable on Windows using PowerShell, run the following command.

```
$Bytes=[IO.File]::ReadAllBytes((Get-Location).ToString()+'\secret.key');
 $env:C3R_SHARED_SECRET=[Convert]::ToBase64String($Bytes)
```

Store key in an environment variable on Linux or macOS

To convert to base64 and set the environment variable on Linux or macOS, run the following command.

```
export C3R_SHARED_SECRET="$(cat secret.key | base64)"
```

Step 7: Encrypt data

To perform this step, you must acquire the AWS Clean Rooms collaboration ID and the shared secret key. For more information, see the Prerequisites.

In the following example, we run the encryption on `ads.csv`, using the schema that we created called `ads.json`.

To encrypt data

1. Store the shared secret key for the collaboration in Step 6: Store the shared secret key in an environment variable.
2. From the command line, enter the following command.

   ```
   java -jar c3r-cli.jar encrypt <name of input .csv file> --schema=<name of schema .json file> --id=<collaboration id> --output=<name of output.csv file> <optional flags>
   ```
3. For *`<name of input .csv file>`*, enter the name of the input .csv file.
4. For `schema=`, enter the name of the .json encryption schema file.
5. For `id=`, enter the collaboration ID.
6. For `output=`, enter the name of the output file (for example, `ads-output.csv`).
7. Include any of the command line flags described in Cryptographic computing parameters and Optional flags in Cryptographic Computing for Clean Rooms.
8. Run the command.

In the example for `ads.csv`, we run the following command.

java -jar c3r-cli.jar encrypt *ads.csv* --schema=*ads.json* --id=*123e4567-e89b-42d3-a456-556642440000* --output=*ads-output.csv*

In the example for **sales.csv**, we run the following command.

java -jar c3r-cli.jar encrypt *sales.csv* --schema=*sales.json* --id=*123e4567-e89b-42d3-a456-556642440000*

> **Note**
>
> In this example, we don't specify an output file name (`--output=sales-output.csv`). As a result, the default output file name `name-of-file.out.csv` was generated.

You are now ready to verify the encrypted data.

Step 8: Verify data encryption

To verify that the data was encrypted

1. View the encrypted data file (for example, `sales-output.csv`).
2. Verify the following columns:
   a. Column **1** – Encrypted (for example, `username_fingerprint`).

      For the fingerprint columns (HMAC), after the version and type prefix (for example, `01:hmac:`), there are 44 characters of base64-encoded data.
   b. Column **2** – Not encrypted (for example, `purchased`).
   c. Column **3** – Encrypted (for example, `product_sealed`).

      For encrypted (SELECT) columns, the length of the cleartext plus any padding after the version and type prefix (for example, `01:enc:`) is directly proportional to the length of the cleartext that was encrypted. That is, the length is the size of the input plus approximately 33 percent overhead because of the encoding.

You are now ready to:

1. Upload the encrypted data to S3.
2. Create an AWS Glue table.

3. Create a configured table in AWS Clean Rooms.

The C3R encryption client will create temporary files that don't contain unencrypted data (unless that data would also be unencrypted in the final output). However, some encrypted values might not be padded properly. Fingerprint columns might contain duplicate values, even if the collaboration setting `allowRepeatedFingerprintValue` is `false`. This issues occurs because the temporary file is written before proper padding lengths and duplicate-removal properties are checked.

If the C3R encryption client fails or is interrupted during encryption, it might stop after writing the temporary file but before checking these properties and deleting the temporary files. Therefore, these temporary files might still be on disk. If this is the case, the contents in these files doesn't protect the plaintext data to the same levels that the output does. In particular, these temporary files might reveal plaintext data to statistical analyses that would not work against the final output. The user should delete these files (particularly a SQLite database) to prevent these files from falling into unauthorized hands.

(Optional) Create a schema (advanced users)

Creating a schema manually is for advanced users.

The following is a description of the JSON schema file format for input files with or without column headers. Advanced users can directly write or modify the schema if desired.

> **Note**
>
> The C3R encryption client can assist you in making a schema through either the interactive process described in Example: Generate an encryption schema with sealed, fingerprint, and cleartext columns or through the creation of a stub template.

Mapped and positional table schemas

The following section describes two kinds of table schemas:

- **Mapped table schema** – This schema is used for encrypting .csv files with a header row and Apache Parquet files.
- **Positional table schema** – This schema is used for encrypting .csv files without a header row.

The C3R encryption client can encrypt a tabular file for a collaboration. To do this, it must have a corresponding schema file that specifies how the encrypted output should be derived from the input.

The C3R encryption client can help generate a schema for an `INPUT` file by running the C3R encryption client schema command at the command line. An example of a command is `java -jar c3r-cli.jar schema --interactive INPUT`.

The schema specifies the following information:

1. Which source columns map to which transformed columns in the output file through their header names (mapped schemas) or position (positional schemas)
2. Which target columns are to remain cleartext
3. Which target columns are to be encrypted for SELECT queries
4. Which target columns are to be encrypted for JOIN queries

This information is encoded in a table-specific JSON schema file, which consists of a single object whose `headerRow` field is a Boolean value. The value must be `true` for Parquet files and .csv files with a header row, and `false` otherwise.

Mapped table schema

The mapped schema has the following shape.

```
{
  "headerRow": true,
  "columns": [
    {
      "sourceHeader": STRING,
      "targetHeader": STRING,
      "type": TYPE,
      "pad": PAD
    },
    ...
  ]
}
```

If `headerRow` is `true`, the next field in the object is `columns`, which contains an array of column schemas that map source headers to target headers (that is, JSON objects describing what the output columns should contain).

- `sourceHeader` – The `STRING` header name of the source column that the data is derived from.

> **Note**
>
> The same source column can be used for multiple target columns.
> A column from the input file not listed as a `sourceHeader` anywhere in the schema doesn't appear in the output file.

- `targetHeader` – The `STRING` header name of the corresponding column in the output file.

> **Note**
>
> This field is optional for mapped schemas. If this field is omitted, the `sourceHeader` is re-used for the header name in the output. Either `_fingerprint` or `_sealed` is appended if the output column is a fingerprint column or sealed column respectively.

- `type` – The `TYPE` of the target column in the output file. That is, one of `cleartext`, `sealed`, or `fingerprint` depending on how the column will be used in the collaboration.
- `pad` – A field of a column schema object that is only present when the `TYPE` is `sealed`. Its corresponding value of `PAD` is an object that describes how the data should be padded before it's encrypted.

```
{
  "type": PAD_TYPE,
  "length": INT
}
```

To specify pre-encryption padding, `type` and `length` are used as follows:

- `PAD_TYPE` as `none` – No padding will be applied to the column's data and the `length` field is not applicable (that is, omitted).
- `PAD_TYPE` as `fixed` – The column's data is padded to the specified `length` of bytes.
- `PAD_TYPE` as `max` – The column's data is padded to the size of the longest value's byte length plus an additional `length` bytes.

The following is an example mapped schema, with a column of each type.

```
{
```

```
  "headerRow": true,
  "columns": [
    {
      "sourceHeader": "FullName",
      "targetHeader": "name",
      "type": "cleartext"
    },
    {
      "sourceHeader": "City",
      "targetHeader": "city_sealed",
      "type": "sealed",
      "pad": {
        "type": "max",
        "length": 16
      }
    },
    {
      "sourceHeader": "PhoneNumber",
      "targetHeader": "phone_number_fingerprint",
      "type": "fingerprint"
    },
    {
      "sourceHeader": "PhoneNumber",
      "targetHeader": "phone_number_sealed",
      "type": "sealed",
      "pad": {
        "type": "fixed",
        "length": 20
      }
    }
  ]
}
```

As a more complex example, the following is an example .csv file with headers.

```
FirstName,LastName,Address,City,State,PhoneNumber,Title,Level,Notes
Jorge,Souza,12345 Mills Rd,Anytown,SC,703-555-1234,CEO,10,
Paulo,Santos,0 Street,Anytown,MD,404-555-111,CIO,9,This is a really long note that
 could really be a paragraph
Mateo,Jackson,1 Two St,Anytown,NY,304-555-1324,COO,9,""
Terry,Whitlock4 N St,Anytown,VA,407-555-8888,EA,7,Secret notes
Diego,Ramirez,9 Hollows Rd,Anytown,VA,407-555-1222,SDE I,4,null
John,Doe,8 Hollows Rd,Anytown,VA,407-555-4321,SDE I,4,Jane's younger brother
```

```
Jane,Doe,8 Hollows Rd,Anytown,VA,407-555-4322,SDE II,5,John's older sister
```

In the following mapped schema example, the columns `FirstName` and `LastName` are `cleartext` columns. The `State` column is encrypted as a `fingerprint` column and as a `sealed` column with a padding of none. The remaining columns are omitted.

```
{
  "headerRow": true,
  "columns": [
    {
      "sourceHeader": "FirstName",
      "targetHeader": "GivenName",
      "type": "cleartext"
    },
    {
      "sourceHeader": "LastName",
      "targetHeader": "Surname",
      "type": "cleartext"
    },
    {
      "sourceHeader": "State",
      "targetHeader": "State_Join",
      "type": "fingerprint"
    },
    {
      "sourceHeader": "State",
      "targetHeader": "State",
      "type": "sealed",
      "pad": {
        "type": "none"
      }
    }
  ]
}
```

The following is the .csv file that results from the mapped schema.

```
givenname,surname,state_fingerprint,state
John,Doe,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv
+1Mk=,01:enc:FQ3n3Ahv9BQQNWQGcugeHzHYzEZE1vapHa2Uu4SRgSAtZ3qObjPA4TcsHt
+BOkMKBcnHWI13BeGG/SBqmj7vKpI=
```

```
Paulo,Santos,01:hmac:CHF4eIrtTNgAooU9v4h9Qjc
+txBnMidQTjdjWuaDTTA=,01:enc:KZ5n5GtaXACco65AXk48BQO2durDNR2ULc4YxmMC8NaZZKKJiksU1IwFadAvV4iBQ1
Mateo,Jackson,01:hmac:iIRnjfNBzryusIJ1w35lgNzeY1RQ1bSfq6PDHW8Xrbk=,01:enc:mLKpS5HIOSgphdEsrzhEd
eN9nBO2gAbIygt4OFn4LalYn9Xyj/XUWXlmn8zFe2T4kyDTD8kGOvpQEUGxAUFk=
Diego,Ramirez,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:rmZhT98Zm
+IIGw1UTjMIJP4IrW/AAltBLMXcHvnYfRgmWP623VFQ6aUnhsb2MDqEw4G5Uwg5rKKZepUxx5uKbfk=
Jorge,Souza,01:hmac:3BxJdXiFFyZ8HBbYNqqEhBVqhNOd7s2ZiKUe7QiTyo8=,01:enc:vVaqWC1VRbhvkf8gnuR7q0z
Terry,Whitlock01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:3c9VEWbODO/
xbQjdGuccLvI7oZTBdPU+SyrJIyr2kudfAxbuMQ2uRdU/q7rbgyJjxZS8M2U35ILJf/lDgTyg7cM=
Jane,Doe,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:9RWv46YLveykeNZ/
G0NdlYFg+AVdOnu05hHyAYTQkPLHnyX+0/jbzD/g9ZT8GCgVE9aB5bV4ooJIXHGBVMXcjrQ=
```

Positional table schema

The positional schema has the following shape.

```
{
  "headerRow": false,
  "columns": [
    [
      {
        "targetHeader": STRING,
        "type": TYPE,
        "pad": PAD
      },
      {
        "targetHeader": STRING,
        "type": TYPE,
        "pad": PAD
      }
    ],
    [],
    ...
  ]
}
```

If `headerRow` is `false`, the next field in the object is `columns`, which contains an array of entries. Each entry is itself an array of zero or more positional column schemas (no `sourceHeader` field), which are JSON objects describing what the output should contain.

- `sourceHeader` – The `STRING` header name of the source column that the data is derived from.

> **Note**
>
> This field must be omitted in positional schemas. In positional schemas, the source column is inferred by the column's corresponding index in the schema file.

- `targetHeader` – The `STRING` header name of the corresponding column in the output file.

> **Note**
>
> This field is required for positional schemas.

- `type` – The `TYPE` of the target column in the output file. That is, one of `cleartext`, `sealed`, or `fingerprint` depending on how the column will be used in the collaboration.
- `pad` – A field of a column schema object that is only present when the `TYPE` is `sealed`. Its corresponding value of `PAD` is an object that describes how the data should be padded before it's encrypted.

```
{
  "type": PAD_TYPE,
  "length": INT
}
```

To specify pre-encryption padding, `type` and `length` are used as follows:

- `PAD_TYPE` as `none` – No padding will be applied to the column's data and the `length` field is not applicable (that is, omitted).
- `PAD_TYPE` as `fixed` – The column's data is padded to the specified `length` of bytes.
- `PAD_TYPE` as `max` – The column's data is padded to the size of the longest value's byte length plus an additional `length` bytes.

> **Note**
>
> `fixed` is useful if you know ahead of time of an upper bound on the byte size of the column's data. An error is raised if any data in that column is longer than the specified `length`.
>
> `max` is convenient when the exact size of input data is unknown because it works regardless of the data's size. However, `max` requires additional processing time because

it encrypts the data twice. `max` encrypts the data once when read in to the temporary file and once after the longest data entry in the column is known.
Also, the length of the longest value isn't saved between invocations of the client. If you plan to encrypt your data in batches, or to encrypt new data periodically, be aware that the resulting ciphertext-lengths might vary among batches.

The following is an example of a positional schema.

```
{
  "headerRow": false,
  "columns": [
    [
      {
        "targetHeader": "name",
        "type": "cleartext"
      }
    ],
    [
      {
        "targetHeader": "city_sealed",
        "type": "sealed",
        "pad": {
          "type": "max",
          "length": 16
        }
      }
    ],
    [
      {
        "targetHeader": "phone_number_fingerprint",
        "type": "fingerprint"
      },
      {
        "targetHeader": "phone_number_sealed",
        "type": "sealed",
        "pad": {
          "type": "fixed",
          "length": 20
        }
      }
    ]
```

```
  ]
}
```

As a complex example, the following is an example .csv file if it didn't have the first row with the headers.

```
Jorge,Souza,12345 Mills Rd,Anytown,SC, 703 -555 -1234,CEO, 10,
Paulo,Santos, 0 Street,Anytown,MD, 404-555-111,CIO, 9,This is a really long note that
 could really be a paragraph
Mateo,Jackson, 1 Two St,Anytown,NY, 304-555-1324,COO, 9, ""
Terry,Whitlock, 4 N St,Anytown,VA, 407-555-8888,EA, 7,Secret notes
Diego,Ramirez, 9 Hollows Rd,Anytown,VA, 407-555-1222,SDE I, 4,null
John,Doe, 8 Hollows Rd,Anytown,VA, 407-555-4321,SDE I, 4,Jane's younger brother
Jane,Doe, 8 Hollows Rd,Anytown,VA, 407-555-4322,SDE II, 5,John's older sister
```

The positional schema has the following form.

```
{
  "headerRow": false,
  "columns": [
    [
      {
        "targetHeader": "GivenName",
        "type": "cleartext"
      }
    ],
    [
      {
        "targetHeader": "Surname",
        "type": "cleartext"
      }
    ],
    [],
    [],
    [
      {
        "targetHeader": "State_Join",
        "type": "fingerprint"
      },
      {
        "targetHeader": "State",
        "type": "sealed",
        "pad": {
```

```
          "type": "none"
        }
      }
    ],
    [],
    [],
    [],
    []
  ]
}
```

The preceding schema produces the following output file with a header row containing the specified target headers.

```
givenname,surname,state_fingerprint,state
Mateo,Jackson,01:hmac:iIRnjfNBzryusIJ1w35lgNzeY1RQ1bSfq6PDHW8Xrbk=,01:enc:ENS6QD3cMVl9vQEGfe9MN
Q8m/Y5SA89dJwKpT5rGPp8e36h6klwDoslpFzGvU0=
Jorge,Souza,01:hmac:3BxJdXiFFyZ8HBbYNqqEhBVqhNOd7s2ZiKUe7QiTyo8=,01:enc:LKo0zirq2+
+XEIIIMNRjAsGMdyWUDwYaum0B+IFP+rUf1BNeZDJjtFe1Z+zbZfXQWwJy52Rt7HqvAb2WIK1oMmk=
Paulo,Santos,01:hmac:CHF4eIrtTNgAooU9v4h9Qjc
+txBnMidQTjdjWuaDTTA=,01:enc:MyQKyWxJ9kvK1xDQQtXlUNwv3F+yrBRr0xrUY/1BGg5KFgOn9pK+MZ7g
+ZNqZEPcPz4lht1u0t/wbTaqzOCLXFQ=
Jane,Doe,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:Pd8sbITBfb0/
ttUB4svVsgoYkDfnDvgkvxzeci0Yxq54rLSwccy1o3/B50C3cpkkn56dovCwzgmmPNwrmCmYtb4=
Terry,Whitlock01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv
+1Mk=,01:enc:Qmtzu3B3GAXKh2KkRYTiEAaMopYedsSdF2e/
ADUiBQ9kv2CxKPzWyYTD3ztmKPMka19dHre5VhUHNpO3O+j1AQ8=
Diego,Ramirez,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:ysdg
+GHKdeZrS/geBIooOEPLHG68MsWpx1dh3xjb+fG5rmFmqUcJLNuuYBHhHAlxchM2WVeV1fmHkBX3mvZNvkc=
John,Doe,01:hmac:UK8s8Cn/WR2JO/To2dTxWD73aDEe2ZUXeSHy3Tv+1Mk=,01:enc:9uX0wZuO7kAPAx
+Hf6uvQownkWqFSKtWS7gQIJSe5aXFquKWCK6yZN0X5Ea2N3bn03Uj1kh0agDWoiP9FRZGJA4=
```

Decrypting data tables with the C3R encryption client

Follow this procedure for collaborations that use Cryptographic Computing for Clean Rooms and the C3R encryption client to encrypt data tables. Use this procedure after you have queried data in the collaboration.

The shared secret key and collaboration ID are required for this procedure.

The member who can receive results decrypts the data using the same shared secret key and collaboration ID that was used to encrypt the data for the collaboration.

> **Note**
>
> AWS Clean Rooms collaborations already limit who can perform and view query results. To perform the decryption, whoever has access to these results needs the same shared secret key and collaboration ID that was used to encrypt the data.

To decrypt an encrypted data table

1. (Optional) View the available commands in the C3R encryption client.
2. (Optional) Navigate to the desired directory and run `ls` (macOS) or `dir` (Windows).
   - Verify that the c3r-cli.jar file and encrypted query results data file are in the desired directory.

     > **Note**
     >
     > If query results are downloaded from the AWS Clean Rooms console interface, they are likely in the **Downloads** folder for your user account. (For example, the **Downloads** folder in your user directory on Windows and macOS.) We recommend that you move the query results file to the same folder as the c3r-cli.jar.
3. Store the shared secret key in the `C3R_SHARED_SECRET` environment variable. For more information, see Step 6: Store the shared secret key in an environment variable.
4. From the AWS Command Line Interface (AWS CLI), run the following command.

   java -jar c3r-cli.jar decrypt *<name of input .csv file>* --id=*<collaboration id>* --output=*<output file name>*
5. Replace each *`user input placeholder`* with your own information:
   a. For `id=`, enter the collaboration ID.
   b. For `output=`, enter the name of the output file (for example, `results-decrypted.csv`).

      If you don't specify an output name, a default name is displayed in the terminal.
   c. View the decrypted data in the specified output file using your preferred CSV or Parquet viewing application (such as Microsoft Excel, a text editor, or other application).

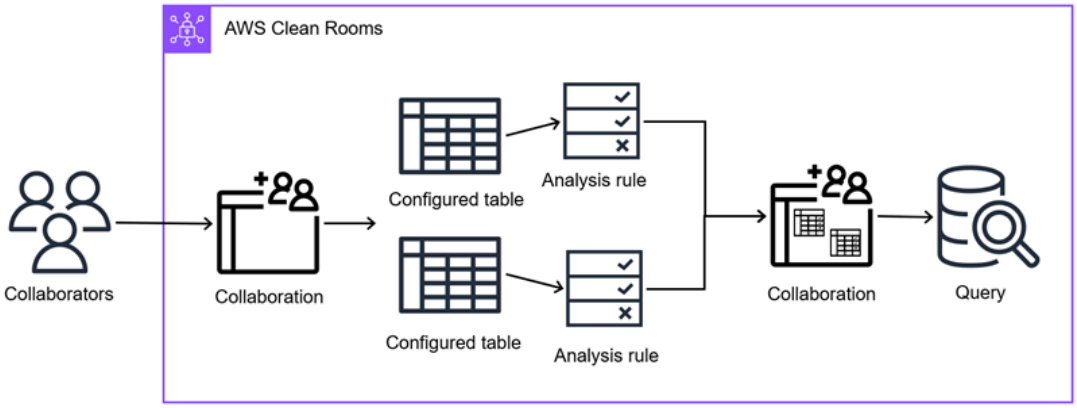
Configured tables in AWS Clean Rooms

A *configured table* is a reference to an existing table in a data source. It contains an analysis rule that determines how the data can be queried in AWS Clean Rooms and can include a data access budget to control table usage. Configured tables can be associated to one or more collaborations.

With AWS Clean Rooms, you can perform aggregation analysis on event data, such as number of purchases compared to number of purchases. You can also perform list analysis on event data, such as enriching overlapping customer data from segment data to CRM data. You can also perform custom queries and set differential privacy on event data, such as viewership data and segment attributes. For any of these analysis types, you can set data access budgets to monitor and control how much of your data is accessed through queries.

First, you create a collaboration in AWS Clean Rooms and add the AWS accounts you want to invite, or join a collaboration you're invited to by creating a membership. Next, you and the other member in the collaboration create configured tables. You both add an analysis rule to the configured tables (aggregation, list, or custom) and optionally set data access budgets. Then, you associate the configured tables to the collaboration. Finally, the member who can query runs a query across the two data tables, consuming the data access budget as queries are executed.

The following diagram summarizes how to work with event data in AWS Clean Rooms.



Topics

- Creating a configured table in AWS Clean Rooms
- Adding an analysis rule to a configured table
- Associating a configured table to a collaboration
- Configuring a data access budget

- Adding a collaboration analysis rule to a configured table
- Configuring differential privacy policy (optional)
- Viewing tables and analysis rules
- Editing a configured table
- Editing configured table tags
- Editing the configured table analysis rule
- Deleting the configured table analysis rule
- Configured table disallowed columns
- Editing configured table associations
- Disassociating configured tables
- Deleting a configured table

Creating a configured table in AWS Clean Rooms

A *configured table* is a reference to an existing table in a data source. It contains an analysis rule that determines how the data can be queried in AWS Clean Rooms. Configured tables can be associated to one or more collaborations.

For information about how to create a configured table using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

Topics

- Creating a configured table – Amazon S3 data source
- Creating a configured table – Amazon Athena data source
- Creating a configured table – Snowflake data source

Creating a configured table – Amazon S3 data source

In this procedure, the member does the following tasks:

- Configures an existing AWS Glue table for use in AWS Clean Rooms. (This step can be done before or after joining a collaboration, unless using Cryptographic Computing for Clean Rooms.)

> **Note**
>
> AWS Clean Rooms supports AWS Glue tables. For more information about getting your data in AWS Glue, see Step 3: Upload your data table to Amazon S3.

- Names the configured table and chooses which columns to use in the collaboration.

The following procedure assumes that:

- The collaboration member has already uploaded their data tables to Amazon S3 and created an AWS Glue table.

> **Note**
>
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

- (Optional) For encrypted data tables only, the collaboration member has already prepared encrypted data tables using the C3R encryption client.

You can use the statistic generation provided by AWS Glue to compute column-level statistics for AWS Glue Data Catalog tables. After AWS Glue generates statistics for tables in the Data Catalog, Amazon Redshift Spectrum automatically uses those statistics to optimize the query plan. For more information about computing column-level statistics using AWS Glue, see Optimizing query performance using column statistics in the *AWS Glue User Guide*. For more information about AWS Glue, see the *AWS Glue Developer Guide*.

To create a configured table – Amazon S3 data source

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. In the upper right corner, choose **Configure new table**.
4. For **Data source**, under **AWS data sources**, choose **Amazon S3**.
5. Under **Amazon S3 table**:

a. Select the **Region** where the S3 table is hosted.

By default, the current Region (such as N. Virginia us-east-1) is selected.

> **Warning**
>
> When your Amazon S3 data source is in a different Region than your processing location, data processing may occur temporarily outside the source Region. Before proceeding, verify that cross-Region data movement complies with your data sovereignty requirements, regulatory compliance policies, and data governance standards.

For more information about Regions, see Regions and Endpoints in the *AWS General Reference*.

b. Choose the **Database** from the dropdown list.

c. Choose the **Table** that you want to configure from the dropdown list.

> **Note**
>
> To verify that this is the correct table, do either one of the following:
>
> - Choose **View in AWS Glue**.
> - Turn on **View schema from AWS Glue** to view the schema.

> **Important**
>
> For AWS Glue tables where the data is in CSV format, the column names and order in the Glue schema must exactly match the CSV data. If they don't align, the allowed columns list for the configured table might not be enforced properly.

6. For **Columns and analysis methods allowed in collaborations**,

a. For **Which columns do you want to allow in collaborations?**

- Choose **All columns** to allow all columns to be queried in the collaboration.

- Choose **Custom list** to allow one or more columns from the **Specify allowed columns** dropdown list to be queried in the collaboration.

b. For **Allowed analysis methods**,

i. Choose **Direct query** to allow SQL queries to be run directly on this table

ii. Choose **Direct job** to allow PySpark jobs to be run directly on this table.

Example

For example, if you want to allow collaboration members to run both direct SQL queries and PySpark jobs on all columns, then choose **All columns**, **Direct query**, and **Direct job**.

7. For **Configured table details**,

a. Enter a **Name** for the configured table.

You can use the default name or rename this table.

b. Enter a **Description** of the table.

The description helps differentiate between other configured tables with similar names.

8. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
9. Choose **Configure new table**.

Now that you have created a configured table, you are ready to:

- Add an analysis rule to the configured table
- Associate the configured table to a collaboration

Creating a configured table – Amazon Athena data source

The Amazon Athena data source option allows you to query data stored in Amazon S3, cataloged in the AWS Glue data catalog or federated catalogs, and access controlled via AWS Lake Formation. Both tables and AWS Glue Data Catalog Views are supported. Lake Formation resource links can be used to share tables and views across AWS accounts and across AWS Regions to the AWS Clean Rooms member account that joins them to an AWS Clean Rooms collaboration.

> **Note**
>
> Only Amazon S3-based datasets can be queried via the Athena data source integration.

In this procedure, the member does the following tasks:

- Configures an existing table or view in the AWS Glue Data Catalog for use the AWS Clean Rooms
- Names the configured table and chooses which columns to use in the collaboration.

The following procedure assumes that:

- The collaboration member has already created the AWS Glue Data Catalog database and table or GDC view.

To create a configured table – Athena data source

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. In the upper right corner, choose **Configure new table**.
4. For **Data source**, under **AWS data sources**, choose **Amazon Athena**.
5. Under **Amazon Athena table**:

   a. Select the **Region** where the Amazon Athena table is hosted.

      By default, the current Region (such as N. Virginia us-east-1) is selected.

      > **Warning**
      >
      > When your Amazon Athena data source is in a different Region than your processing location, data processing may occur temporarily outside the source Region. Before proceeding, verify that cross-Region data movement complies with your data sovereignty requirements, regulatory compliance policies, and data governance standards.

For more information about Regions, see Regions and Endpoints in the *AWS General Reference*.

b. Choose the **Catalog** from the dropdown list.

By default, **AWS Glue Data Catalog** is selected.

- **AWS Glue Data Catalog** – The default catalog for tables in AWS Glue.
- **Federated catalog** – Available if you've configured AWS Glue Catalog Federation to connect to remote Apache Iceberg REST catalogs. For more information, see Catalog federation in the *AWS Lake Formation Developer Guide*.

c. Choose the **Database** from the dropdown list.

d. Choose the **Table** that you want to configure from the dropdown list.

> **Note**
>
> To verify that this is the correct table, do either one of the following:
>
> - Choose **View in AWS Glue** or **View in AWS Lake Formation** (depending on your catalog type).
> - Turn on **View schema from AWS Glue** to view the schema.

6. For **Amazon Athena configurations**,

a. Choose a **Workgroup** from the dropdown list.

b. For **S3 output location**, choose a recommended action, based on one of the following scenarios.

Scenario	Recommended action
Your **Workgroup** doesn’t have a default output location.	Enter the **S3 output location** or choose **Browse S3**.
Your **Workgroup** enforces your default output location.	The **S3 output location** is automatically chosen and you can't change it.

Scenario	Recommended action
Your **Workgroup** doesn't enforce your default output location.	Enter the **S3 output location** or choose **Browse S3**.

7. For **Columns allowed in collaborations**, choose an option based on your goal.

Your goal	Recommended option
Allow all columns for use in AWS Clean Rooms (subject to analysis rules)	**All columns**
Allow one or more columns from the **Specify allowed columns** dropdown list	**Custom list**

8. For **Configured table details**,
   a. Enter a **Name** for the configured table.

      You can use the default name or rename this table.
   b. Enter a **Description** of the table.

      The description helps differentiate between other configured tables with similar names.
   c. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
9. Choose **Configure new table**.

Now that you have created a configured table, you are ready to:

- Add an analysis rule to the configured table
- Associate the configured table to a collaboration

Creating a configured table – Snowflake data source

In this procedure, the member does the following tasks:

- Configures an existing Snowflake table for use in AWS Clean Rooms. (This step can be done before or after joining a collaboration, unless using Cryptographic Computing for Clean Rooms.)

- Names the configured table and chooses which columns to use in the collaboration.

The following procedure assumes that:

- The collaboration member has already uploaded their data tables to Snowflake.
- (Optional) For encrypted data tables only, the collaboration member has already prepared encrypted data tables using the C3R encryption client.

To create a configured table – Snowflake data source

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. In the upper right corner, choose **Configure new table**.
4. For **Data source**, under **Third-party clouds and data sources**, choose **Snowflake**.
5. Specify the **Snowflake credentials** using an existing secret ARN or storing a new secret for this table.

   Use existing secret ARN

   1. If you have a secret ARN, enter it in the **Secret ARN** field.

      You can look up your secret ARN by choosing **Go to AWS Secrets Manager**.
   2. If you have an existing secret from another table, choose **Import secret ARN from existing table**.

   > **Note**
   >
   > The secret ARN can be cross-account.

   Store a new secret for this table

   1. Enter the following Snowflake credentials:
      - **Snowflake username**
      - **Snowflake warehouse**

- **Snowflake role**
- **Snowflake Privacy Enhanced Mail (PEM) private key**

2. For encryption, do one of the following:
   - To use the AWS managed key (default), leave the **Customize encryption settings** checkbox cleared.
   - To use a custom AWS KMS key:
     - Select the **Customize encryption settings** checkbox.
     - For **KMS key**, enter the key ARN or choose one from the list.
3. Enter a **Secret name** to help you find your credentials later.

6. For **Snowflake table and schema details**, enter the details manually or automatically import the details.

   Enter the details manually

   1. Enter the **Snowflake account identifier**.

      For more information, see Account identifiers in the Snowflake documentation.

      Your account identifier must be in the format used for Snowflake drivers. You need to replace the period (.) with a hyphen (-) so the identifier is formatted as **`<orgname>-<account_name>`**.
   2. Enter the **Snowflake database**.

      For more information, see Snowflake database in the Snowflake documentation.
   3. Enter the **Snowflake schema name**.
   4. Enter the **Snowflake table name**.

      For more information, see Understanding Snowflake Table Structures in the Snowflake documentation.
   5. For the **Schema**, enter the **Column name** and choose the **Data type** from the dropdown list.
   6. Choose **Add column** to add more columns.
      - If you choose an **Object data type**, specify the **Object schema**.

Example object schema

```
name STRING,
location OBJECT(
    x INT,
    y INT,
    metadata OBJECT(uuid STRING)
),
history ARRAY(TEXT)
```

- If you choose an **Array data type**, specify the **Array schema**.

  Example array schema

  ```
  OBJECT(x INT, y INT)
  ```

- If you choose a **Map data type**, specify the **Map schema**.

  Example map schema

  ```
  STRING, OBJECT(x INT, y INT)
  ```

Automatically import the details

1. Export your COLUMNS view from Snowflake as a CSV file.

   For more information about the Snowflake COLUMNS view, see COLUMNS view in the Snowflake documentation.

2. Choose **Import from file** to import the CSV file and specify any additional information.

   The database name, schema name, table name, column names and data types are automatically imported.

   - If you choose an **Object data type**, specify the **Object schema**.
   - If you choose an **Array data type**, specify the **Array schema**.
   - If you choose a **Map data type**, specify the **Map schema**.

3. Enter the **Snowflake account identifier**.

   For more information, see Account identifiers in the Snowflake documentation.

> **Note**
>
> Only S3 tables cataloged in AWS Glue can be used to retrieve the table schema automatically.

7. For **Columns allowed in collaborations**, choose an option based on your goal.

Your goal	Recommended option
Allow all columns for use in AWS Clean Rooms (subject to analysis rules)	**All columns**
Allow one or more columns from the **Specify allowed columns** dropdown list	**Custom list**

8. For **Configured table details**,
   a. Enter a **Name** for the configured table.

      You can use the default name or rename this table.

   b. Enter a **Description** of the table.

      The description helps differentiate between other configured tables with similar names.

   c. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
9. Choose **Configure new table**.

Now that you have created a configured table, you are ready to:

- Add an analysis rule to the configured table
- Associate the configured table to a collaboration

Adding an analysis rule to a configured table

The following sections describe how to add an analysis rule to your configured table. By defining the analysis rules, you can authorize the member who can query to run queries that match a specific analysis rule supported by AWS Clean Rooms.

AWS Clean Rooms supports the following types of analysis rules:

- Aggregation analysis rule
- List analysis rule
- Custom analysis rule in AWS Clean Rooms

There can be only one analysis rule per configured table. You can configure the analysis rule any time before you associate your configured tables with the collaboration.

> **Important**
>
> If you are using Cryptographic Computing for Clean Rooms and have encrypted data tables in the collaboration, the analysis rule you add to the encrypted configured table should be consistent with how the data was encrypted. For example, if you encrypted the data for SELECT (aggregation analysis rule), you shouldn't add the analysis rule for JOIN (list analysis rule).

Topics

- Adding an aggregation analysis rule to a table (guided flow)
- Adding a list analysis rule to a table (guided flow)
- Adding a custom analysis rule to a table (guided flow)
- Adding analysis rule to a table (JSON editor)
- Next steps

Adding an aggregation analysis rule to a table (guided flow)

The *aggregation analysis rule* allows queries that aggregate statistics without revealing row-level information using COUNT, SUM, and AVG functions along optional dimensions.

This procedure describes the process of adding an aggregation analysis rule to your configured table by using the **Guided flow** option in the AWS Clean Rooms console.

> **Note**
>
> Configured tables using non-S3 data sources only support custom analysis rules.

To add the aggregation analysis rule to a table (guided flow)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table.
4. On the configured table detail page, choose **Configure analysis rule**.
5. Under **Step 1: Choose analysis rule type**, under **Analysis rule type**, choose the **Aggregation** option.
6. Under **Creation method**, select **Guided flow**, and then choose **Next**.
7. Under **Step 2: Specify query controls**, for **Aggregate functions**:

   a. Choose an **Aggregate function** from the dropdown:

      - **COUNT**
      - **COUNT DISTINCT**
      - **SUM**
      - **SUM DISTINCT**
      - **AVG**

   b. Choose which columns can be used in the **Aggregate function** from the **Columns** dropdown.

   c. (Optional) Choose **Add another function** to add another aggregate function and associate one or more columns to that function.

      > **Note**
      >
      > At least one aggregate function is required.

d. (Optional) Choose **Remove** to remove an aggregate function.

8. For **Join controls**,

   a. Choose one option for **Allow table to be queried by itself**:

If you choose...	Then ...
No, only overlap can be queried	The table can be queried only when joined to a table owned by the member who can query.
Yes	The table can be queried by itself or when joined to other tables.

   b. Under **Specify join columns**, choose the columns that you want to allow to be used in the INNER JOIN statement.

      This is *optional* if you have selected **Yes** in the previous step.

   c. Under **Specify allowed operators for matching**, choose which, if any, operators can be used for matching on multiple join columns. If you select two or more JOIN columns, one of these operators is required.

If you choose...	Then ...
AND	You can include `AND` in the `INNER JOIN` match conditions to join one column to another column between tables.
OR	You can include `OR` in the `INNER JOIN` match conditions to combine multiple column matches between tables. This logical operator is useful for obtaining a higher match rate.

9. *(Optional)* For **Dimension controls**, in the **Specify dimension columns** dropdown, choose which columns you want to allow to be used in the SELECT statement, and the WHERE, GROUP BY, and ORDER BY parts of the query.

> **Note**
>
> Aggregate function or join columns can't be used as **Dimension** columns.

10. For **Scalar functions**, choose one option for **Which scalar functions do you want to allow?**

If you choose...	Then ...
All currently supported by AWS Clean Rooms	You allow all scalar functions currently supported by AWS Clean Rooms. • You can choose **View list** to see the entire list of **Scalar functions supported in AWS Clean Rooms**.
A custom list	You can customize which scalar functions to allow. • Choose one or more options from the **Specify allowed scalar functions** dropdown.
None	You don't want to allow any scalar functions .

For more information, see Scalar functions.

11. Choose **Next**.
12. Under **Step 3: Specify query results controls**, for **Aggregation constraints**:
    a. Select the dropdown list for each **Column name**.
    b. Select the dropdown list for each **Minimum number of distinct values** that must be met for each output row to be returned, after the COUNT DISTINCT function is applied to it.
    c. Choose **Add constraint** to add more aggregation constraints.
    d. (Optional) Choose **Remove** to remove an aggregation constraint.
13. For **Additional analyses applied to output**, select an option based on your goal.

Your goal	Recommended option
Allow only direct queries on this table. Deny additional analyses from being run on query results. The table can only be used for direct querying.	**Not allowed**
Allow but don't require both direct queries and additional analyses on this table.	**Allowed**
Require that the table can only be used in direct queries that are processed with one of the required additional analyses. Direct queries on this table must be further processed before they can be returned.	**Required**

14. Choose **Next**.
15. Under **Step 4: Review and configure**, review the selections you've made for the previous steps, edit if necessary, and then choose **Configure analysis rule**.

You see a confirmation message that you've successfully configured an aggregation analysis rule to the table.

Adding a list analysis rule to a table (guided flow)

The *list analysis rule* allows queries that output row-level lists of the overlap between the associated table and a table of the member who can query.

This procedure describes the process of adding the list analysis rule to your configured table using the **Guided flow** option in the AWS Clean Rooms console.

> **Note**
>
> Configured tables using non-S3 data sources only support custom analysis rules.

To add a list analysis rule to a table (guided flow)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table.
4. On the configured table detail page, choose **Configure analysis rule**.
5. Under **Step 1: Choose analysis rule type**, under **Analysis rule type**, choose the **List** option.
6. Under **Creation method**, select **Guided flow**, and then choose **Next**.
7. Under **Step 2: Specify query controls**, for **Join controls**:
   a. Under **Specify join columns**, choose the columns that you want to allow to be used in the INNER JOIN statement.
   b. Under **Specify allowed operators for matching**, choose which, if any, operators can be used for matching on multiple join columns. If you select two or more JOIN columns, one of these operators is required.

If you choose...	Then ...
AND	You can include `AND` in the `INNER JOIN` match conditions to join one column to another column between tables.
OR	You can include `OR` in the `INNER JOIN` match conditions to combine multiple column matches between tables. This logical operator is useful for obtaining a higher match rate.

8. *(Optional)* For **List controls**, in the **Specify list columns** dropdown, choose which columns you want to allow to be used in the query output (that is, used in the SELECT statement), or used to filter results (that is, the WHERE statement).
9. Choose **Next**.
10. Under **Step 3: Specify query results controls**, for **Additional analyses applied to output**, select an option based on your goal.

Your goal	Recommended option
Allow only direct queries on this table. Deny additional analyses from being run on query results. The table can only be used for direct querying.	**Not allowed**
Allow but don't require both direct queries and additional analyses on this table.	**Allowed**
Require that the table can only be used in direct queries that are processed with one of the required additional analyses. Direct queries on this table must be further processed before they can be returned.	**Required**

11. Under **Step 4: Review and configure**, review the selections you've made for the previous steps, edit if necessary, and then choose **Configure analysis rule**.

You see a confirmation message that you've successfully configured a list analysis rule for the table.

Adding a custom analysis rule to a table (guided flow)

The custom analysis rule enables custom SQL queries or PySpark jobs on a configured table. The custom analysis rule is required if you're using:

- Analysis templates to allow a specific set of pre-approved SQL queries or PySpark jobs or a specific set of accounts that can provide queries that use your data.
- AWS Clean Rooms Differential Privacy to protect against user-identification attempts.
- Non-S3 data sources, such as Amazon Athena or Snowflake.

This procedure describes the process of adding the custom analysis rule to your configured table using the **Guided flow** option in the AWS Clean Rooms console.

To add a custom analysis rule to a table (guided flow)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table.
4. On the configured table detail page, choose **Configure analysis rule**.
5. Under **Step 1: Choose analysis rule type**, under **Analysis rule type**, choose the **Custom** option.
6. Under **Creation method**, select **Guided flow**, and then choose **Next**.
7. Under **Step 2: Specify analysis controls**, for **Direct analysis controls**, choose an option based on your goal.

Your goal	Recommended action
Review each new analysis before it is allowed to be run on this configured table	1. Under **Analysis templates allowed to be run**, choose **Add analysis template**. 2. Choose the appropriate **Collaboration** and the **Analysis template** from the dropdown lists. 3. Choose **Next**.
Allow specific collaborators to run any analyses of a chosen type without review on this table	1. Under **Analysis type**, a. Choose **Any query** to allow any query created by the AWS account you specify. b. Choose **Any query** to allow any job created by the AWS account you specify. 2. Under **AWS accounts allowed to create any analysis**, choose **Add AWS account**. 3. Enter an AWS account or choose the an **AWS account ID**. from the dropdown list.

Your goal	Recommended action
	4. (Optional) Choose **Add another AWS account** to add another AWS account. 5. Choose **Next**.

8. Under **Step 3: Specify analysis results controls**,

   a. For **Job results controls**, note that no additional results controls are supported.

   b. Under **Query results controls**, for **Columns not allowed in output**, choose the columns you want to be allowed in the query output, based on your goal.

Your goal	Recommended action
Allow all columns to be returned in query outputs	1. Choose **None** 2. Proceed to **Additional analyses applied to output**.
Disallow certain columns from being returned in query outputs	1. Choose **Custom list** 2. Under **Specify disallowed columns**, choose the columns that you want removed from query outputs.

   c. For **Additional analyses applied to output** choose whether additional analyses can be applied to the query output, based on your goal.

Your goal	Recommended option
• Allow only direct queries on this table. • Deny additional analyses from being run on query results. • The table can only be used for direct querying.	**Not allowed**
Allow but don't require both direct queries and additional analyses on this table.	**Allow**

Your goal	Recommended option
• Require that the table can only be used in direct queries that are processed with one of the required additional analyses. • Direct queries on this table must be further processed before they can be returned.	**Required**

d. Choose **Next**.

9. (Optional) Under **Step 4: Set differential privacy**, determine whether you want differential privacy turned on or off.

   Differential privacy is a mathematically-proven technique to protect your data from re-identification attacks.

   > **Note**
   >
   > AWS Clean Rooms Differential Privacy is only available for collaborations where the data is stored in Amazon S3.

   For **Differential privacy**, choose whether to turn differential privacy on or off, based on your goal.

Your goal	Recommended action
• You don't require protection against re-identification attempts • Your table doesn't have user-level data	1. Choose **Turn off**. 2. Choose **Next**.
• You require protection against re-identi fication attempts • Your table has user-level data	1. Choose **Turn on**. 2. Select the **User identifier column** that contains the unique identifier of your users, such as the `user_id` column, whose privacy you want to protect.

Your goal	Recommended action
	To turn on differential privacy for two or more tables in a collaboration, you must configure the same column as the **User identifier column** in both analysis rules to maintain a consistent definitio n of users across tables. In case of a misconfiguration, the member who can query receives an error message that there are two columns to choose from in order to compute the number of user contributions (for example, the number of ad impressions made by a user) while running the query. 3. Choose **Next**.

10. Under **Step 5: Review and configure**, review the selections you've made for the previous steps, edit if necessary, and then choose **Configure analysis rule**.

You see a confirmation message that you've successfully configured a custom analysis rule for the table.

Adding analysis rule to a table (JSON editor)

The following procedure shows how to add an analysis rule to a table using the **JSON editor** option in the AWS Clean Rooms console.

> **Note**
>
> Configured tables using non-S3 data sources only support custom analysis rules.

To add an aggregation, list, or custom analysis rule to a table (JSON editor)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

2. In the left navigation pane, choose **Tables**.
3. Choose the configured table.
4. On the configured table detail page, choose **Configure analysis rule**.
5. Under **Step 1: Choose analysis rule type**, under **Analysis rule type**, choose either the **Aggregation**, **List**, or **Custom** option.
6. Under **Creation method**, select **JSON editor**, and then choose **Next**.
7. Under **Step 2: Specify controls**, you can choose to insert a query structure (**Insert template**) or insert a file (**Import from file**).

If you choose...	Then ...
Insert template	1. Specify the parameters for the selected analysis rule in the **Analysis rule definitio n**. 2. You can press **Ctrl** + **Spacebar** to enable auto-complete. For more information about aggregation analysis rule parameters, see Aggregation analysis rule - query controls. For more information about list analysis rule parameters, see List analysis rule - query controls.
Import from file	1. Select your JSON file from your local drive. 2. Choose **Open**. The **Analysis rule definition** displays the analysis rule from the uploaded file.

8. Choose **Next**.
9. Under **Step 3: Review and configure**, review the selections you've made for the previous steps, edit if necessary, and then choose **Configure analysis rule**.

You receive a confirmation message that you've successfully configured an analysis rule for the table.

Next steps

Now that you configured an analysis rule to your configured table, you are ready to:

- Associate a configured table to a collaboration
- Query the data tables (as a member who can query)

Associating a configured table to a collaboration

After you have created a configured table and added an analysis rule to it, you can associate it to a collaboration and give AWS Clean Rooms a service role to access your AWS Glue tables.

> **Note**
>
> This service role has permissions to the tables. The service role is assumable only by AWS Clean Rooms to run allowed queries on behalf of the member who can query. No collaboration members (other than the data owner) have access to the underlying tables in the collaboration. The data owner can turn on differential privacy to make their tables available for querying by other members.

Data access budget

When you associate a configured table, you can apply a data access budget. A *data access budget* controls how many times a table can be used for queries, jobs, and ML input channels in a collaboration. These budgets help organizations manage resource utilization and control costs by limiting table use.

Each time a table is used in a query, job, or ML input channel, the budget for that table is reduced by one. When the budget reaches zero, the table can't be used in SQL queries, Pyspark jobs, nor as part of ML input channels derived from the table.

You can establish a per period budget that refreshes periodically, a lifetime budget for overall usage, or both. By default, table usage is unlimited.

- Per period budget – A renewable allocation that limits the amount of times this table can be used within a specified time period. You can set the period to daily, weekly, or monthly. This budget can be set to automatically refresh on a daily, weekly, or monthly basis.
- Lifetime budget – A running allocation that limits the total amount of times this table can be used.

Associate a configured table

The following topics describe how to associate a configured table and apply a data access budget to a collaboration using the AWS Clean Rooms console.

For information about how to associate your configured tables to the collaboration using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

Step 1: Complete the prerequisites

To associate a configured table, you must complete the following prerequisites:

- An AWS Glue table that points to an Amazon S3 folder location (not a single file)
- For encrypted AWS Glue tables:
  - A service role with permissions to use AWS KMS keys for decrypting AWS Glue tables
  - For AWS KMS-encrypted Amazon S3 datasets: The service role must also have permissions to use the AWS KMS key to decrypt Amazon S3 data

For information about configuring encryption, see Setting up encryption in AWS Glue in the *AWS Glue Developer Guide*.

To verify your AWS Glue table location:

1. Open the AWS Glue console at https://console.aws.amazon.com/glue/
2. View your table details and confirm the location points to an S3 folder

Step 2: Associate a configured table

To associate a configured table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. Choose the method to associate your table:
   a. From the configured table detail page:
      i. In the left navigation pane, choose **Tables**.
      ii. Choose the configured table.
      iii. On the configured table detail page, choose **Associate to collaboration**.
      iv. For the **Associate table to collaboration** dialog box, choose the **Collaboration** from the dropdown list.
   b. From the collaboration detail page:
      i. In the left navigation pane, choose **Collaborations**.
      ii. Choose the collaboration.
      iii. On the **Tables** tab, choose **Associate table**.
3. On the **Associate table** page, do one of the following:
   - Choose an existing configured table – Choose the **Configured table name** that you want to associate with the collaboration from the dropdown list.
   - Configure a new table – Choose **Configure new table** and follow the prompts on the **Configure new table** page.
   - View the schema and analysis rule for the configured table – Turn on **View schema and analysis rule**.
4. For **Table association details**,
   a. Enter a **Name** for the associated table.

      You can use the default name or rename this table.
   b. (Optional) Enter a **Description** of the table.

      The description helps with writing queries.

5. Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

> **Note**
>
> If you are associating a configured table backed by Amazon Athena, choose an **Existing service role name** from the dropdown list. Ensure the service role has IAM and, if needed, Lake Formation permissions to the dataset.

If you choose...	Then ...
Create and use a new service role	• AWS Clean Rooms creates a service role with the required policy for this table. • The default **Service role name** is `cleanrooms-<timestamp>` • You must have permissions to create roles and attach policies. • If your input data is encrypted, you can select **This data is encrypted with a KMS key** and then enter an AWS KMS key that will be used to decrypt your data input.

If you choose...	Then ...
Use an existing service role	1. Choose an **Existing service role name** from the dropdown list. The list of roles are displayed if you have permissions to list roles. If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use. 2. View the service role by choosing the **View in IAM** external link. If there are no existing service roles, the option to **Use an existing service role** is unavailable. By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions. 3. (Optional) Select the **Add a pre-configured policy with necessary permissions to this role** check box to add attach necessary permissions to the role. You must have permissions to modify roles and create policies.

Note

- AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.

- If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
- If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

6. If you want to enable **Configured table association tags** for the configured table association resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
7. Choose **Next**.
8. On the **Configure collaboration analysis rule** page, choose one of the following:
   - **Yes, create a collaboration analysis rule now** – Associates your table with this collaboration and creates a collaboration analysis rule
   - **No, I will create a collaboration analysis rule later** – Associates your table with this collaboration only. You can create a collaboration analysis rule later.
9. If you choose **Yes, create a collaboration analysis rule now**, for **Results delivery**, choose the **Members allowed to receive results for query output** from the dropdown list.
10. Choose **Next**.
11. On the **Add data access budget** page, for **Data access budget configuration**, choose one of the following:
    - **Yes, add a data access budget now** – Associates your table with this collaboration and adds a data access budget. You can select either a period budget, a lifetime budget, or both.
    - **No, I will add a data access budget later** – Associates your table with this collaboration only. You can add a data access budget later.

      If you select **No, I will add a data access budget later**, skip to step 15.
12. If you choose **Yes, add a data access budget now**, choose one of the following budget configurations:

Per period budget only	Lifetime budget only	Both per period and lifetime budgets
1. Leave **Add per period budget** selected.	1. Clear **Add per period budget**.	1. Leave **Add per period budget** selected.

Per period budget only	Lifetime budget only	Both per period and lifetime budgets
2. Enter a **Per period budget amount** between 1 and 1,000,000. 3. For **Period**, choose **Daily**, **Weekly**, or **Monthly**. 4. (Optional) Leave **Automatically refresh budget weekly** selected to renew the allocation. 5. Clear **Add lifetime budget**.	2. Select **Add lifetime budget**. 3. Enter a **Lifetime budget amount** between 1 and 1,000,000.	2. Enter a **Per period budget amount** between 1 and 1,000,000. 3. For **Period**, choose **Daily**, **Weekly**, or **Monthly**. 4. Leave **Automatically refresh budget weekly** selected. 5. Select **Add lifetime budget**. 6. Enter a **Lifetime budget amount** between 1 and 1,000,000.

13. Review your selections under **Data access budget summary**.

    Example

    For example, if you've chosen a **Per period budget amount** of 1,000, set the **Period** to **Weekly**, left the **Automatically refresh budget weekly** checkbox selected, and set the **Lifetime budget** to 1,000,000, then the **Access budget summary** will display the following message: Every week, this table can be used up to 1,000 times for running queries or jobs. This budget is set to automatically refresh every Sunday at 00:00 UTC, and will continue to refresh until this table has reached its lifetime budget of 1,000,000 uses.

14. (Optional) If you want to enable **Data access budget tags** for the access budget resource, choose **Add new tag** and enter a Key and Value pair.
15. Choose **Next**.
16. Review the information on the **Review and create** page.
    a. If you need to edit any sections, choose **Edit**.
    b. Edit your configurations, and then choose **Next**.
17. Choose **Associate table**.

Step 3: Next steps

Now that you associated your configured data table to the collaboration, you are ready to:

- Add a collaboration analysis rule to the configured table
- Edit the collaboration, if you're the collaboration creator
- Query the data tables (as a member who can query)

Configuring a data access budget

A collaborator can view, add, edit, and delete a *data access budget* to set a limit on the number of times a table can be used in queries, PySpark jobs, and ML jobs. Use these budgets to manage data and costs.

Each time a table is queried, a PySpark job is run on the table, or an ML job is run using an ML input channel derived from a table, the budget for that table is reduced by one. When the budget reaches zero, you can't run queries, PySpark jobs, or ML jobs on that table.

You can establish a per period budget that refreshes periodically, a lifetime budget for overall usage, or both. By default, table usage is unlimited.

- Per period budget – A renewable allocation that limits the amount of times this table can be used within a specified time period. You can set the period to daily, weekly, or monthly. This budget can be set to automatically refresh on a daily, weekly, or monthly basis.
- Lifetime budget – A running allocation that limits the total amount of times this table can be used.

Topics

- Adding a data access budget to an existing associated table
- Viewing a data access budget
- Editing a data access budget
- Deleting a data access budget

Adding a data access budget to an existing associated table

As a collaboration member, you can add a data access budget to an existing associated table.

To add a data access budget to an existing associated table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Select the option button next to the table you want to add a data access budget to.
6. From the **Actions** dropdown list, under **Data access budget**, select **Add** (if there isn't already a budget).
7. Choose one of the following budget configurations:

Per period budget only	Lifetime budget only	Both per period and lifetime budgets
1. Leave **Add per period budget** selected. 2. Enter a **Per period budget amount** between 1 and 1,000,000. 3. For **Period**, choose **Daily**, **Weekly**, or **Monthly**. 4. (Optional) Leave **Automatically refresh budget weekly** selected to renew the allocation. 5. Clear **Add lifetime budget**.	1. Clear **Add per period budget**. 2. Select **Add lifetime budget**. 3. Enter a **Lifetime budget amount** between 1 and 1,000,000.	1. Leave **Add per period budget** selected. 2. Enter a **Per period budget amount** between 1 and 1,000,000. 3. For **Period**, choose **Daily**, **Weekly**, or **Monthly**. 4. Leave **Automatically refresh budget weekly** selected. 5. Select **Add lifetime budget**. 6. Enter a **Lifetime budget amount** between 1 and 1,000,000.

8. Review your selections under **Data access budget summary**.

9. **Example**

   For example, if you've chosen a **Per period budget amount** of 1,000, set the **Period** to **Weekly**, left the **Automatically refresh budget weekly** checkbox selected, and set the **Lifetime budget** to 1,000,000, then the **Access budget summary** will display the following message: Every week, this table can be used up to 1,000 times for running queries or jobs. This budget is set to automatically refresh every Sunday at 00:00 UTC, and will continue to refresh until this table has reached its lifetime budget of 1,000,000 uses.

10. (Optional) If you want to enable **Data access budget tags** for the access budget resource, choose **Add new tag** and enter a Key and Value pair.

11. Choose **Add data access budget**.

Viewing a data access budget

You can view a data access budget from the **Tables** tab or from the table details page.

To view a data access budget

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Do one of the following:

   - Under the **Remaining data access budget** column, select the budget to view the details.
   - Choose a table, and on the table details page, scroll down to view the **Data access budget details** section.

Editing a data access budget

As a collaboration member, you can edit the data access budget. When you edit a data access budget, it resets the current budget balance.

You can edit a data access budget from the **Tables** tab or from the table details page.

Tables tab

To edit a data access budget from the Tables tab

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Select the option button next to the table you want to edit.
6. From the **Actions** dropdown list, under **Data access budget**, select **Edit**.
7. On the **Edit data access budget** page, update the **Per period budget** or **Lifetime budget** information.
8. View the **Data access budget summary** to verify that the edits you've made are correct.
9. Choose **Save changes**.

Table details page

To edit a data access budget from the table details page

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Choose a table.
6. On the table details page, scroll down to the **Data access budget details** section.
7. From the **Actions** dropdown list, choose **Edit**.
8. On the **Edit data access budget** page, update the **Per period budget** or **Lifetime budget** information.
9. Choose **Save changes**.

Deleting a data access budget

You can delete a data access budget from the **Tables** tab or from the table details page.

Tables tab

To delete a data access budget from the Tables tab

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Select the option button next to the table you want to delete.
6. From the **Actions** dropdown list, under **Data access budget**, select **Delete**.

   > **Important**
   >
   > You can't undo this action and your data access budget will be reset to unlimited.

7. If you're certain that you want to delete the data access budget, choose **Delete**.

Table details page

To delete a data access budget from the table details page

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Choose a table.
6. On the table details page, scroll down to the **Data access budget details** section.
7. From the **Actions** dropdown list, choose **Delete**.

> **Important**
>
> You can't undo this action and your data access budget will be reset to unlimited.

8. If you're certain that you want to delete the data access budget, choose **Delete**.

Adding a collaboration analysis rule to a configured table

The *collaboration analysis rule* allows you to specify controls that are specific to this collaboration. These controls work together with the configured table analysis rule to determine how this table can be analyzed within this collaboration.

You add a collaboration analysis rule to a configured table after you've created a configured table, added an analysis rule, and associated it to a collaboration. You need to add a collaboration analysis rule if the table is configured to support direct analysis or to allow additional analysis.

- **Direct analysis** – The table can be used in queries that analyze it directly. For example, in a query that outputs an aggregate measurement analysis or a list of identifiers for activation.
- **Additional analysis** – The table can also be used as input into additional analyses, in addition to queries that analyze it directly. For example, the table can be used in a query that is a seed for a lookalike ML model, or an ML input channel for a custom ML model.

To add the collaboration analysis rule to a table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Tables** tab, under **Tables associated by you**, view the configured table you've associated to the collaboration.
   - If **Direct analysis status** or **Additional analysis status** has a status of **Ready**, then the table is ready to be queried.
   - If **Direct analysis status** or **Additional analysis status** has a status of **Not ready**, then select the status, and then choose **Configure** in the dialog box.

5. On the **Configure collaboration analysis rule** page, expand **View configured table analysis rule** to view the details.
6. For **Allowed additional analyses**, choose the option based on your goal.

Your goal	Recommended option
Allow any additional analyses on the table.	**Any**
Allow only additional analyses on the table by a specific member.	**Any by specific members**
Allow only specific analyses on the table.	**Custom list**

7. For **Results delivery**, specify who can receive results from the **Members allowed to receive results for query output** dropdown.
8. Choose **Configure analysis rule**.

Configuring differential privacy policy (optional)

> **Note**
>
> AWS Clean Rooms Differential Privacy is only available for collaborations where the data is stored in Amazon S3.

This procedure describes the process of configuring the differential privacy policy in a collaboration by using the **Guided flow** option in the AWS Clean Rooms console. This is a one-time step for all tables with differential privacy protection.

To configure differential privacy settings (guided flow)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Tables** tab of the collaboration page, choose **Configure differential privacy policy**.
5. On the **Configure differential privacy policy** page, choose values for the following properties:

- **Privacy budget**
- **Refresh privacy budget monthly**
- **Noise added per query**

You can use the default values or enter custom values that support your specific use case. After choosing values for **Privacy budget** and **Noise added per query**, you can preview the resulting utility in terms of the number of aggregations that are possible across all queries on your data.

6. Choose **Configure**.

You'll see a confirmation message that you've successfully configured the differential privacy policy for the collaboration.

Now that you configured differential privacy, you are ready to:

- Query the data tables (as a member who can query)
- Collaborations (if you're the collaboration creator)
- Use differential privacy with intermediate tables

Viewing differential privacy usage logs

As a collaboration member who is protecting data with differential privacy, after you have created a collaboration with differential privacy, you can monitor the usage of the privacy budget.

To view how many aggregations were run and how much of the privacy budget was used

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Choose **View usage logs** (blue text).
6. View the usage details, including the privacy budget and how much utility was provided.

Editing a differential privacy policy

At any time after configuring the differential privacy policy, you can update it to better reflect your privacy needs.

To edit the differential privacy policy

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Tables** tab of the collaboration page, under **Tables associated by you**, choose **Edit**.
5. On the **Edit differential privacy** page, choose new values for the following properties:

   - **Privacy budget** – Move the slider bar to either increase or decrease the budget at any point during a collaboration. You can't decrease the budget after the member who can query has started querying your data. If the **Privacy budget** is increased, AWS Clean Rooms will continue using the existing budget until it is fully consumed before utilizing the newly added privacy budget.
   - **Noise added per query** – Move the slider bar to either increase or decrease the **Noise added per query** at any point during a collaboration.

   > **Note**
   >
   > You can chose **Interactive examples** to explore how different values of **Privacy budget** and **Noise added per query** affect the number of aggregate functions that you can run.

   You can't change the value of the **Privacy budget refresh**. To change your selection, you must delete the differential privacy policy and create a new one.
6. Choose **Save changes**.

You see a confirmation message that you've successfully edited the differential privacy policy.

Deleting a differential privacy policy

You can delete the differential privacy policy from the **Tables** tab of a collaboration.

To delete the differential privacy policy

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Tables** tab of the collaboration page, next to **Differential privacy policy**, select **Delete**.
5. If you're certain that you want to delete the differential privacy policy, choose **Delete**.

After deleting a differential privacy policy, you can't access the privacy budget usage logs from that policy. Tables with differential privacy turned on can't be queried if the differential privacy policy is deleted.

Viewing the calculated differential privacy parameters

For users with expertise in differential privacy, you can view the calculated differential privacy parameters from the **Analysis** tab of a collaboration.

To view the calculated differential privacy parameters

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Analysis** tab, in the **Results** section, select **View calculated differential privacy parameters**.

In the **Calculated differential privacy parameters** table, you can see sensitivity values of aggregate functions, which is defined as the maximum amount by which the result of a function can change if a single user's records are added, removed, or modified. The list includes the following differential privacy parameters:

- **User contribution limit** (UCL) is the maximum number of rows contributed by a user in a SQL query. For example, if you want to count the total number of matched impressions in a specified campaign where each user can have multiple impressions, AWS Clean Rooms Differential Privacy needs to bound the number of impressions of a single user in order to ensure that the differential privacy calculation is accurate. In other words, if any user has more impressions than the bound, then AWS Clean Rooms automatically takes a uniform random sample of that user's impressions as per the computed UCL value and exclude the remaining impressions of that user while executing the query. The UCL value equals to 1 if you are counting the number of unique users. This is because adding, removing, or modifying a single user can change the count of distinct users by at most 1.
- **Minimum value** is the lower bound of an expression used within an aggregate function such as `sum()`. For example, if the expression is a column known as `purchase_value`, minimum value is the lower bound of the column.
- **Maximum value** is the upper bound of an expression used within an aggregate function such as `sum()`. For example, if the expression is a column known as `purchase_value`, maximum value is the upper bound of the column.

In the **Calculated differential privacy parameters** table, you can use these parameters to better understand the total amount of noise in query results. For example, when the configured **Noise added per query** is 30 users and a `COUNT DISTINCT (user_id)` query is run, then AWS Clean Rooms Differential Privacy adds random noise that falls between -30 and 30 with high probability because the sensitivity of `COUNT DISTINCT` is 1. In the case of a `COUNT` query with the same configuration, AWS Clean Rooms Differential Privacy adds statistical noise that is scaled by the user contribution limit because a single user could contribute multiple rows to the query result. In the case of `SUM` query like `SUM (purchase_value)` where all the column values are positive, the total noise is scaled by the user contribution limit times the maximum value. AWS Clean Rooms Differential Privacy automatically computes the sensitivity parameters to perform noise addition at query run-time and depletes the privacy budget. The depletion of privacy budget is required because the sensitivity parameters are data-dependent.

Viewing tables and analysis rules

To view tables associated with the collaboration and the analysis rules

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. Choose one of the following:
   a. To view your tables associated in the collaboration, for **Tables associated by you**, choose a table (blue text).
   b. To view other tables associated in the collaboration, for **Tables associated by collaborators**, choose a table (blue text).
6. View the table details and analysis rules on the table details page.

Editing a configured table

Prerequisites:

- An AWS account with access to AWS Clean Rooms

The following sections explain how to edit the name, description, and configuration details of tables for Amazon S3, Amazon Athena, and Snowflake data sources.

For information about how to edit a configured table using the AWS SDKs, see the *AWS Clean Rooms API Reference*.

To edit a configured table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table that you created.
4. On the configured table detail page, choose **Edit**.
5. Edit your configuration.
6. Choose **Save changes**.

Editing configured table tags

As a collaboration member, after you have created a configured table, you can manage the tags on the configured table resource on the **Configured tables** tab.

To edit the configured table tags

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table that you created.
4. On the configured table detail page, scroll down to the **Tags** section.
5. Choose **Manage tags**.
6. On the **Manage tags** page, you can do the following:
   - To remove a tag, choose **Remove**.
   - To add a tag, choose **Add new tag**.
   - To save your changes, choose **Save changes**.

Editing the configured table analysis rule

To edit the configured table analysis rule

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table that you created.
4. On the configured table detail page, scroll down to either the **Aggregation analysis rule**, **List analysis rule**, or the **Custom analysis rule** section. (Your choice depends on which type of analysis rule you chose for the configured table.)
5. Choose **Edit**.
6. On the **Edit analysis rule** page, you can:
   - Modify the **Analysis rule definition** by:
     - Modifying the JSON editor.

    - Choosing **Import from file** to upload a new analysis rule definition.
  - Preview what members will see in a collaboration by selecting from the following options:
    - **Table view**
    - **JSON**
    - **Example query**
7. Choose **Save changes** to save your changes.

Deleting the configured table analysis rule

> **Warning**
>
> This action can't be undone and impacts all related resources.

To delete the configured table analysis rule

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. Choose the configured table that you created.
4. On the configured table detail page, scroll down to either the **Aggregation analysis rule**, **List analysis rule**, or the **Custom analysis rule** section. (Your choice depends on which type of analysis rule you chose for the configured table.)
5. Choose **Delete**.
6. If you're certain that you want to delete the analysis rule, choose **Delete**.

Configured table disallowed columns

The disallowed output columns configuration is a control in the AWS Clean Rooms custom analysis rule that enables you to define the list of columns (if any) that you don't allow to be projected in the query result. The columns referenced in this list are considered "disallowed output columns". This means that any reference to such column through transformation, aliasing, or other means may not be present in the final SELECT (projection) of the query.

While the capability prohibits columns from being directly projected in the output, it doesn't fully prevent underlying values from being indirectly inferred through other mechanisms. These columns can still be used in a projection clause (such as in a subquery or a Common Table Expression (CTE)), as long as they aren't referenced in the very final projection.

The disallowed output columns configuration gives you the flexibility to apply and codify control on your table in combination with analysis template level reviews based on use cases and corresponding privacy requirements.

For more information on how to set this configuration, see Adding a custom analysis rule to a table (guided flow).

> **Disallowed output columns query constraint and CACHE TABLE**
>
> The disallowed output columns constraint in the custom analysis rule is enforced on cached tables. A cached table cannot reference a disallowed output column in its SELECT clause. To use a column with a disallowed output column constraint in a subsequent part of your query, convert the cached table to a common table expression (CTE).

Examples

The following examples display how the disallowed output columns control is applied.

- Member A is in a collaboration with Member B.
- Member B is member who can run queries.
- Member A defines a table *users* with the columns *age*, *gender*, *email*, and *name*. The columns *age* and *name* are disallowed output columns.
- Member B defines a table *pets* with a similar set of columns *age*, *gender*, and *owner_name*. However, they don't set any constraints on the output columns, meaning that all columns in the table can be projected freely in the query.

If Member B runs the following query, it's blocked because disallowed output columns can't be directly projected:

```
SELECT
  age
```

```
FROM
  users
```

If Member B runs the following query, it's blocked because disallowed output columns can't be implicitly projected via project star:

```
SELECT
  *
FROM
  users
```

If Member B runs the following query, it's blocked because transformations of disallowed output columns can't be projected:

```
SELECT
  COUNT(age)
FROM
  users
```

If Member B runs the following query, it's blocked because disallowed output columns can't be referenced in final projection using an alias:

```
SELECT
  count_age
FROM
  (SELECT COUNT(age) AS count_age FROM users)
```

If Member B runs the following query, it's blocked because transformed restricted columns are projected in output:

```
SELECT
  CONCAT(name, email)
FROM
  users
```

If Member B runs the following query, it's blocked because disallowed output columns defined in CTE can't be referenced in the final projection:

```
WITH cte AS (
  SELECT
    age AS age_alias
  FROM
    users
)
SELECT age_alias FROM cte
```

If Member B runs the following query, it's blocked because disallowed output columns can't be used as sort or partition keys in the final projection:

```
SELECT
  LISTAGG(gender) WITHIN GROUP (ORDER BY age) OVER (PARTITION BY age)
FROM
  users
```

If Member B runs the following query, it succeeds because columns that are part of the disallowed output columns can still be used across other constructs in the query, such as in join or filter clauses.

```
SELECT
  u.name,
  p.gender,
  p.age
FROM
  users AS u
JOIN
  pets AS p
ON
  u.name = p.owner_name
```

In the same scenario, Member B can also use the *name* column in *users* as a filter or sort key:

```
SELECT
  u.email,
```

```
  u.gender
FROM
  users AS u
WHERE
  u.name = 'Mike'
ORDER BY
  u.name
```

Additionally, the disallowed output columns from *users* can be used in intermediate projections such as subqueries and CTEs, such as:

```
WTIH cte AS (
 SELECT
   u.gender,
   u.id,
   u.first_name
 FROM
   users AS u
)
SELECT
  first_name
FROM
  (SELECT cte.gender, cte.id, cte.first_name FROM cte)
```

Editing configured table associations

As a collaboration member, you can edit the configured table associations that you have created.

To edit configured table associations

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose **Tables** tab.
5. For **Tables associated by you**, choose a table.
6. On the table details page, scroll down to view the **Table association details**.

7. Choose **Edit**.
8. On the **Edit configured table associations** page, update the **Description** or the **Service access information**.
9. Choose **Save changes**.

Disassociating configured tables

As a collaboration member, you can disassociate a configured table from the collaboration. This action prevents the member who can query from querying the table.

> **Warning**
>
> Disassociating a configured table from a collaboration causes all dependent intermediate tables (and their descendants) to become unusable with a status of `BASE_TABLE_REMOVED`. The stored data in those intermediate tables is removed and storage-based billing stops. For more information, see Deleting an intermediate table.

To disassociate a configured table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose **Tables** tab.
5. For **Tables associated by you**, select the option button next to the table that you want to disassociate.
6. Choose **Disassociate**.
7. In the dialog box, confirm the decision to disassociate the configured table and prevent the member who can query from querying the table by choosing **Disassociate**.

Deleting a configured table

You can delete a configured table that you own. This action cannot be undone and affects all related resources.

> **Warning**
>
> All collaborations to which the configured table is associated will be impacted. When you delete a configured table, all of its associations stop being queryable in the respective collaborations. All analyses that refer to this table fail.
>
> Deleting a configured table does not delete your configured table associations or data stored in dependent intermediate tables in collaborations. For more information about cleaning up intermediate tables, see Deleting an intermediate table.

To delete a configured table

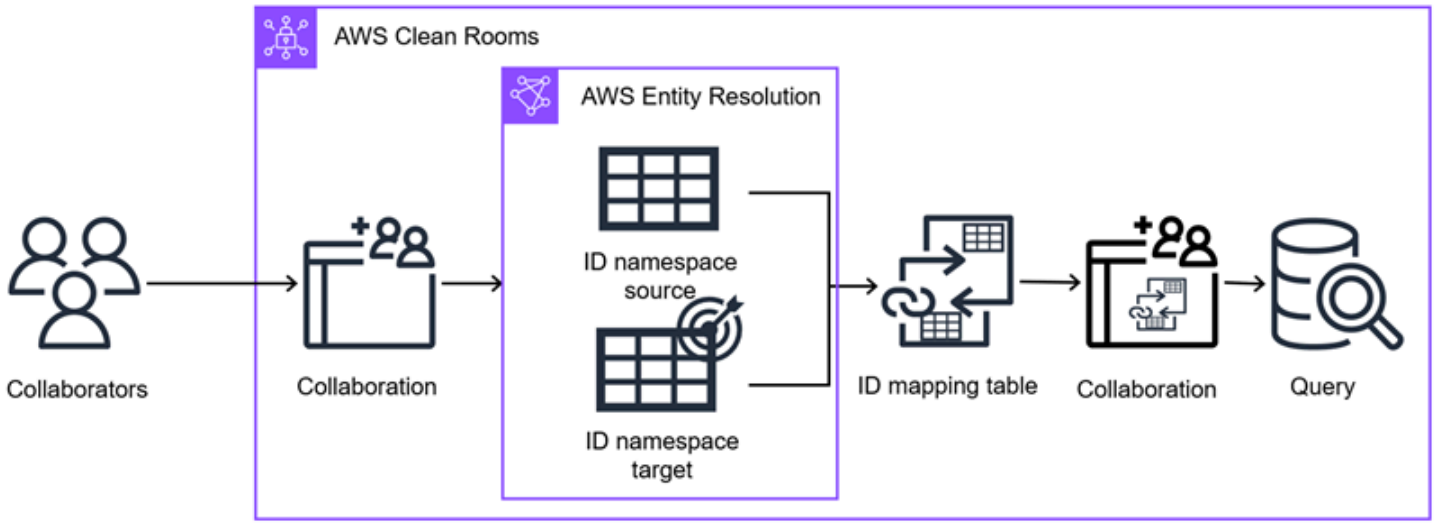
1. Open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Configured tables**.
3. Choose the configured table that you want to delete.
4. Choose **Delete**.
5. In the confirmation dialog, review the warning that all collaborations to which the table is associated will be impacted.
6. To confirm the deletion, enter **`confirm`** in the text field, and then choose **Delete**.

AWS Entity Resolution in AWS Clean Rooms

With AWS Entity Resolution in AWS Clean Rooms, you can translate data from a source to a target, populate an ID mapping table with the translated data, and query the data.

First, you create a collaboration in AWS Clean Rooms and add the AWS accounts you want to invite, or join a collaboration you're invited to by creating a membership. Next, you perform ID mapping on two data tables. You do this by either associating an existing ID namespace source or creating a new one in AWS Entity Resolution. The other member of the collaboration associates an existing ID namespace target or creates a new ID namespace target. Then, you create and populate an ID mapping table from the two associated ID namespaces. Finally, the member who can query runs a query across the two data tables by joining on the ID mapping table.

The following diagram summarizes how to work with AWS Entity Resolution in AWS Clean Rooms.



> **Note**
>
> The currently supported transcoding service provider is LiveRamp, which is available in the following AWS Regions: US East (N. Virginia), US East (Ohio), and US West (Oregon).

Topics

- ID namespaces in AWS Clean Rooms
- ID mapping tables in AWS Clean Rooms

ID namespaces in AWS Clean Rooms

An *ID namespace* is a wrapper around your identity table that enables you to provide metadata explaining your dataset and how to use it in an ID mapping workflow. An *ID mapping workflow* is a data processing job that maps data from an input data source to an input data target based on the specified ID mapping method. It produces an ID mapping table.

There are two types of ID namespaces: **Source** and **Target**. The **Source** contains configurations for the source data that will be processed in an ID mapping workflow. The **Target** contains a configuration of the target data which all sources will resolve to. To define the input data that you want to resolve across two AWS accounts, create an ID namespace source and an ID namespace target to translate your data from one set (**Source**) to another (**Target**).

You can either create a new ID namespace or you can associate an existing one. For more information about how to create an ID namespace in AWS Entity Resolution, see Define input data using an ID namespace in the *AWS Entity Resolution User Guide*.

Topics

- Creating and associating a new ID namespace
- Associating an existing ID namespace
- Editing ID namespace associations
- Disassociating ID namespace associations

Creating and associating a new ID namespace

Each member of the collaboration must create and associate either an ID namespace **Source** or an ID namespace **Target** before creating an ID mapping table to query identity data.

If you have already created an ID namespace in AWS Entity Resolution, skip to Associating an existing ID namespace.

To create and associate a new ID namespace

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.

4. On the **Entity resolution** tab, choose **Associate ID namespace**.
5. On the **Associate ID namespace** page, for **Entity resolution data**, choose **Create ID namespace**.

   The AWS Entity Resolution console appears in a new tab.
6. Follow the prompts on the **Create ID namespace** page in the AWS Entity Resolution console.

   a. For **Details**, enter the **ID namespace name**, **Description**, and select the **ID namespace type** (either **Source** or **Target**).

   b. For the **ID namespace method**, choose either the **Rule-based** method for rule-based matching or **Provider services** for third-party transcoding.

   c. Specify the **Data input type**, depending on the ID namespace method you've chosen.

   d. Choose **Create ID namespace**.
7. Go back to the AWS Clean Rooms console.
8. On the **Associate ID namespace** page, for **Entity resolution data**, choose the **AWS Entity Resolution ID namespace** source or target that you want to associate with the collaboration from the dropdown list.
9. For **Association details**, take the following steps.

   a. Enter a **Name** for the associated ID namespace.

      You can use the default name or rename this ID namespace.

   b. (Optional) Enter a **Description** of the ID namespace.

      The description helps with writing queries.
10. Specify the **AWS Clean Rooms access** permissions by selecting an option and then taking the recommended action.

Option	Recommended action
Allow AWS Clean Rooms to add and manage permission policy	AWS Clean Rooms creates a service role with the required policy for this association.
Add and manage permissions manually	Do one of the following: • Review the **Resource policy** and add necessary permissions to the policy.

Option	Recommended action
	• Use an existing policy by choosing **Add policy statement**. You must have permissions to modify roles and create policies. **Note** If you can't modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

11. (Optional) For **Advanced ID mapping table configurations**, modify the default protections for the column that comes from the ID namepsace.

    The ID mapping table is configured by default to only allow an `INNER JOIN` on both the `sourceID` column and the `targetID` column. You can modify this configuration so that the column that comes from this ID namespace (either `sourceID` or `targetID`) can be allowed anywhere in the query.

Your goal	Recommended option
Categorize the column as a "join column" and only allow it in an `INNER JOIN` clause	**Yes**
Categorize the column as a "dimension column" and allow it anywhere in the query, including a `JOIN` clause, `SELECT`, `WHERE` and `GROUP BY` statements of the query.	**No, allow anywhere in the query**

12. (Optional) If you want to enable **Tags** for the ID namepsace resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
13. Choose **Associate**.

14. On the **Entity resolution** tab, under the **Associated ID namespaces** table, view the associated ID namespace and verify that the ID namespace type is correct (**Source** or **Target**).

After all members in the collaboration have associated their ID namespaces, you can create an ID mapping table and query the data.

Associating an existing ID namespace

In this procedure, each member associates either their existing ID namespace source or their ID namespace target in the collaboration.

To associate an existing ID namespace

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Entity resolution** tab, choose **Associate ID namespace**.
5. On the **Associate ID namespace** page, for **Entity resolution data**, choose the **AWS Entity Resolution ID namespace** source or target that you want to associate with the collaboration from the dropdown list.
6. For **Association details**, take the following steps.

   a. Enter a **Name** for the associated ID namespace.

      You can use the default name or rename this ID namespace.

   b. (Optional) Enter a **Description** of the ID namespace.

      The description helps with writing queries.
7. Specify the **AWS Clean Rooms access** permissions by selecting an option and then taking the recommended action.

Option	Recommended action
Allow AWS Clean Rooms to add and manage permission policy	AWS Clean Rooms creates a service role with the required policy for this association.

Option	Recommended action
Add and manage permissions manually	Do one of the following: • Review the **Resource policy** and add necessary permissions to the policy. • Use an existing policy by choosing **Add policy statement**. You must have permissions to modify roles and create policies. **Note** If you can't modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

8. (Optional) For **Advanced ID mapping table configurations**, modify the default protections for the column that comes from the ID namepsace.

   The ID mapping table is configured by default to only allow an `INNER JOIN` on both the `sourceID` column and the `targetID` column. You can modify this configuration so that the column that comes from this ID namespace (either `sourceID` or `targetID`) can be allowed anywhere in the query.

Your goal	Recommended option
Categorize the column as a "join column" and only allow it in an `INNER JOIN` clause.	**Yes**
Categorize the column as a "dimension column" and allow it anywhere in the query, including a `JOIN` clause, `SELECT`, `WHERE`, and `GROUP BY` statements of the query.	**No, allow anywhere in the query**

9. (Optional) If you want to enable **Tags** for the ID namepsace resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
10. Choose **Associate**.
11. On the **Entity resolution** tab, under the **Associated ID namespaces** table, view the associated ID namespace and verify that the ID namespace type is correct (**Source** or **Target**).

After all members in the collaboration have associated their ID namespaces, you can create an ID mapping table and query the data.

Editing ID namespace associations

As a collaboration member, you can edit the ID namespace associations that you have created.

To edit an ID namespace association

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Entity resolution** tab.
5. For **Associated ID namespaces**, choose an ID namespace.
6. On the ID namespace details page, scroll down to view the **ID namespace association details**.
7. Choose **Edit**.
8. On the **Edit ID namespace associations** page, edit any of the following:

    a. For **Association details**, update the **Name** or the **Description**.

    b. (Optional) For **Advanced ID mapping table configurations**, modify the default protections for the column that comes from the ID namepsace.

    The ID mapping table is configured by default to only allow an `INNER JOIN` on both the `sourceID` column and the `targetID` column. You can modify this configuration so that the column that comes from this ID namespace (either `sourceID` or `targetID`) can be allowed anywhere in the query.

Your goal	Recommended option
Categorize the column as a "join column" and only allow it in an `INNER JOIN` clause	**Yes**
Categorize the column as a "dimensio n column" and allow it anywhere in the query, including a `JOIN` clause, `SELECT`, `WHERE` and `GROUP BY` statements of the query.	**No, allow anywhere in the query**

9. Choose **Save changes**.

Disassociating ID namespace associations

As a collaboration member, you can disassociate an ID namespace from the collaboration. This action prevents the member who can query from querying the table.

> **Warning**
>
> Disassociating an ID namespace association from a collaboration deletes any data from derived ID mapping tables, rendering them not queryable.
> For example, if your ID namespace association was used as a SOURCE in three different ID mapping tables, then all the data from these ID mapping tables will be deleted when you disassociate your ID namespace association.
> Disassociating an ID namespace association also causes all dependent intermediate tables (and their descendants) to become unusable with a status of `BASE_TABLE_REMOVED`. The stored data in those intermediate tables is removed and storage-based billing stops. For more information, see Deleting an intermediate table.

To disassociate an ID namespace association

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.

3. Choose the collaboration.
4. Choose the **Entity resolution** tab.
5. For **Associated ID namespaces**, select the option button next to the ID namespace that you want to disassociate.
6. Choose **Disassociate**.
7. In the dialog box, confirm your decision to disconnect the ID namespace by choosing **Disassociate**. This action prevents any member who can query from accessing the ID mapping table.

   If a member of the collaboration removes one of the ID namespaces, you can't repopulate the ID mapping table if the source has left the collaboration.

   Even though the ID mapping table was populated previously, disassociating the ID namespace means you can no longer run queries on that table.

ID mapping tables in AWS Clean Rooms

An *ID mapping table* is a resource in AWS Clean Rooms that enables multiparty identity mapping in a collaboration.

Before you create an ID mapping table, you must first have both source and target data configured as ID namespaces.

After you create an ID mapping table, you use an ID mapping workflow to translate the source ID namespace to the target ID namespace. You can do this using either a rule-based method, or a provider service transcoding method.

An *ID mapping workflow* is a data processing job that maps data from an input data source to an input data target based on the specified ID mapping workflow method. This workflow populates an ID mapping table.

> **Note**
>
> ID mapping tables can only be created from datasets that are stored in Amazon S3 and crawled into AWS Glue tables.

There are two ID mapping workflow methods: rule-based ID mapping or provider services ID mapping:

- Rule-based ID mapping – You use matching rules to translate first-party data from a source to a target.
- Provider services ID mapping – You use the LiveRamp provider service to translate third-party data from a source to a target.

> **Note**
>
> The currently supported transcoding service provider is LiveRamp. Any member in the collaboration who has an subscription with LiveRamp through AWS Data Exchange can create the ID mapping table. If you already have a subscription to LiveRamp, but not through AWS Data Exchange, contact LiveRamp to get a private offer. For more information, see Subscribe to a provider service on AWS Data Exchange in the *AWS Entity Resolution User Guide*.

Topics

- Creating and populating a new ID mapping table
- Populating an existing ID mapping table
- Editing an ID mapping table
- Deleting an ID mapping table

Creating and populating a new ID mapping table

Prerequisites

Before creating an ID mapping table, ensure you have:

- An associated ID namespace source and target
- The ID namespaces configured for either rule-based ID mapping or provider services ID mapping

You can create two types of ID mapping tables:

- Rule-based – Uses matching rules to translate first-party data

- Provider services – Uses LiveRamp to translate RampIDs

After you create an ID mapping table, you can either populate it immediately by running an ID mapping workflow, or you can wait to populate the table later.

After the ID mapping table is successfully populated, you can then run a multi-table join query on the ID mapping table to join the `sourceId` with the `targetId` and analyze the data.

Topics

- Create an ID mapping table (rule-based)
- Create an ID mapping table (provider services)

Create an ID mapping table (rule-based)

This topic describes the process of creating an ID mapping table that uses matching rules to translate first-party data from a source to a target.

When you create a rule-based ID mapping table, you can choose to process only new, updated, or deleted records in the workflow by turning on incremental processing.

To create and populate a new ID mapping table using the rule-based method

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the navigation pane, choose **Collaborations**.
3. Choose the collaboration, and then choose the **Entity resolution** tab.
4. Choose **Create ID mapping table**.
5. Under **ID mapping settings**, do one of the following:
   - To create a new workflow, keep **Create a new ID mapping workflow** selected.
   - To use an existing workflow, clear the checkbox and select a workflow from the dropdown list, then skip to step 9.
6. Under **Identity data**, view or configure the source and target.
   - For a single ID namespace pair: Review the pre-selected **Source** and **Target**.
   - For multiple ID namespaces: Select the **Source** and **Target** from the dropdown lists.

7. (Optional) Select **Turn on incremental processing** to process only new, updated, or deleted records in the workflow.

   AWS Entity Resolution processes only new, updated, or deleted records in either the Source or Target ID namespace, rather than recreating the entire ID mapping table.

   If you leave this option unselected, AWS Entity Resolution runs the default batch processing ID mapping workflow on the ID mapping table.
8. Under **Rule parameters**, configure the following:

   - **Rule controls** – Choose whether the **Target** or **Source** provides matching rules.

     You can view the rules by turning on **Show rules**.

     Rule controls must be compatible between the source and the target ID namespace to be used in an ID mapping workflow. For example, if a source ID namespace limits rules to the target but the target ID namespace limits rules to the source, this results in an error.
   - **Comparison type** is automatically set to **Multiple input fields.**

     This is because both participants had selected this option previously.
   - **Record matching**
     - **One source to one target** – Stores one matching record per target
     - **Many sources to one target** – Stores all matching records per target

     > **Note**
     >
     > The limitations specified for the source and target ID namespaces must be compatible.
9. For **ID mapping details**, configure the following:

   a. Enter an **ID mapping table name** or keep the default name.
   b. (Optional) Enter a **Description** of the ID mapping table.

      The description helps with writing queries.
10. For **AWS Clean Rooms access**, choose one:

    - **Allow AWS Clean Rooms to add and manage permission policy** – Creates a service role automatically.

- **Add and manage permissions manually** – Either review and modify the resource policy or choose **Add policy statement**.

> **Note**
>
> If you can't modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

11. For **AWS Entity Resolution access**, choose one:

    This section is only visible if you're creating a new ID mapping table.

    - **Create and use a new service role**
      - The default Service role name is `entityresolution-id-mapping-workflow-<timestamp>`
      - (Optional) For encrypted data, select **This data is encrypted by a KMS key** and enter the **AWS KMS key**.
    - **Use an existing service role**
      - Choose an **Existing service role name** from the dropdown list or enter a role ARN.

        The list of roles are displayed if you have permissions to list roles.

        View the service role by choosing the **View in IAM** external link.

        If there are no existing service roles, the option to **Use an existing service role** is unavailable.

        By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.
      - (Optional) Select **Add a pre-configured policy with necessary permissions to this role** to attach necessary permissions to the role.

        You must have permissions to modify roles and create policies.

12. (Optional) Under **Additional settings**, configure:

    a. **ID mapping table settings**

       - To enable custom encryption, choose **Customize encryption settings** and enter an AWS KMS key.

> **Note**
>
> This KMS key needs to grant the required permissions to use within AWS Entity Resolution to `cleanrooms.amazonaws.com` using a KMS key policy. For more details about the required permissions for working with encryptions with an ID mapping workflow, see Create a workflow job role for AWS Entity Resolution in the *AWS Entity Resolution User Guide*.

- To add tags, choose **Add new tag** and enter key-value pairs

b. **ID mapping workflow settings** (new workflows only):

- To use different names, clear **Keep the same ID mapping table name and description** and enter new values.
- To add tags, choose **Add new tag** and enter key-value pairs

13. Choose one of the following:

- **Create ID mapping table** – Creates an empty table you can populate later (Populating an existing ID mapping table)
- **Create and populate ID mapping table** – Creates and immediately populates the table (may take several hours)

The ID mapping workflow process begins. During this process, the ID mapping table is populated with translated IDs. The ID mapping workflow might take a few hours to process.

After the ID mapping table is successfully populated, you can query the ID mapping table to join the `sourceId` with the `targetId` and analyze the data.

Create an ID mapping table (provider services)

This topic describes the process of creating an ID mapping table that uses a provider service (LiveRamp). The LiveRamp provider services translates a set of source RampIDs to another using either maintained or derived RampIDs.

To create a new ID mapping table using the provider services method

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the navigation pane, choose **Collaborations**.
3. Choose the collaboration, and then choose the **Entity resolution** tab.
4. Choose **Create ID mapping table**.
5. Under **ID mapping settings**, do one of the following:
   - To create a new workflow, keep **Create a new ID mapping workflow** selected.
   - To use an existing workflow, clear the checkbox and select a workflow from the dropdown list, then skip to step 9.
6. Under **Identity data**, view or configure the source and target.
   - For a single ID namespace pair: Review the pre-selected **Source** and **Target**.
   - For multiple ID namespaces: Select the **Source** and **Target** from the dropdown lists.
7. Under **Method**, verify that the selected ID mapping workflow method is **LiveRamp transcoding**.
8. For **LiveRamp configurations**, do one of the following:
   - Enter the **LiveRamp ID manager ARN** and **LiveRamp secret manager ARN**.
   - Choose **Import from existing workflow**.
9. For **ID mapping details**, configure the following:
   a. Enter an **ID mapping table name** or keep the default name.
   b. (Optional) Enter a **Description** of the ID mapping table.

      The description helps with writing queries.
10. For **AWS Clean Rooms access**, choose one:
    - **Allow AWS Clean Rooms to add and manage permission policy** – Creates a service role automatically.
    - **Add and manage permissions manually** – Either review and modify the resource policy or choose **Add policy statement**.

> **Note**
>
> If you can't modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

11. For **AWS Entity Resolution access** choose one:

    This section is only visible if you're creating a new ID mapping table.

    - **Create and use a new service role**
      - The default Service role name is `entityresolution-id-mapping-workflow-<timestamp>`
      - (Optional) For encrypted data, select **This data is encrypted by a KMS key** and enter the **AWS KMS key**.
    - **Use an existing service role**
      - Choose an **Existing service role name** from the dropdown list or enter a role ARN.

        The list of roles are displayed if you have permissions to list roles.

        View the service role by choosing the **View in IAM** external link.

        If there are no existing service roles, the option to **Use an existing service role** is unavailable.

        By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.
      - (Optional) Select **Add a pre-configured policy with necessary permissions to this role** to attach necessary permissions to the role.

        You must have permissions to modify roles and create policies.

12. (Optional) Under **Additional settings** configure:

    a. **ID mapping table settings**

       - To enable custom encryption, choose **Customize encryption settings** and enter an AWS KMS key.

> **Note**
>
> This KMS key needs to grant the required permissions to use within AWS Entity Resolution to `cleanrooms.amazonaws.com` using a KMS key policy. For more details about the required permissions for working with encryptions with an ID mapping workflow, see Create a workflow job role for AWS Entity Resolution in the *AWS Entity Resolution User Guide*.

- To add tags, choose **Add new tag** and enter key-value pairs.

b. **ID mapping workflow settings** (new workflows only):

- To use different names, clear **Keep the same ID mapping table name and description** and enter new values.
- To add tags, choose **Add new tag** and enter key-value pairs.

13. Choose one of the following:

- **Create ID mapping table** – Creates an empty table you can populate later (Populating an existing ID mapping table)
- **Create and populate ID mapping table** – Creates and immediately populates the table (may take several hours)

The ID mapping workflow process begins. During this process, the ID mapping table is populated with translated IDs. The ID mapping workflow might take a few hours to process.

After the ID mapping table is successfully populated, you can query the ID mapping table to join the `sourceId` with the `targetId` and analyze the data.

Populating an existing ID mapping table

When new data is added to an ID namespace, use this workflow. If you have chosen to turn on incremental processing when you created the ID mapping table, you have the ability to processes only new, updated, or deleted records in either the Source or Target ID namespace, rather than recreating the entire ID mapping table.

To populate an existing ID mapping table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Go to the **Entity resolution** tab.
5. Under the **ID mapping tables** section, select an ID mapping table.
6. If you haven't turned on incremental processing for this ID mapping table, choose an ID mapping table and then choose **Populate**.
7. If you have turned on incremental processing for this ID mapping table, choose **Populate with** and then choose one of the following:

   - **Incremental processing** – Processes only new, updated, or deleted records in either the Source or Target ID namespace.

     Recommended for frequent updates, daily runs, or real-time data synchronization.
   - **Batch processing** – Processes the entire ID mapping table.

     Recommended for initial setup, periodic full refreshes, or when significant changes occur in both Source and Target ID namespaces.
   - **Delete only processing** – Processes only deleted records from the Source ID namespace and updates the Target ID namespace accordingly.

     Recommended for quickly synchronizing removals.
8. The ID mapping workflow process begins.

   During this process, the ID mapping table is populated with transcoded IDs. The ID mapping workflow might take a few hours to process.

After the ID mapping table is successfully populated, you can query the ID mapping table to join the `sourceId` with the `targetId`.

Editing an ID mapping table

As a collaboration member, you can edit the ID mapping table that you have created.

To edit an ID mapping table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Entity resolution** tab.
5. For **ID mapping tables**, choose a table.
6. On the ID mapping table details page, scroll down to view the **ID mapping table details**.
7. Choose **Edit**.
8. On the **Edit ID mapping table** page, update the **Description** or the **Service access information**.
9. Choose **Save changes**.

Deleting an ID mapping table

As a collaboration member, you can delete an ID mapping table that you have created. This action prevents the member who can query from querying the table.

> **Warning**
>
> Deleting a mapping table permanently removes any populated data.
> Deleting an ID mapping table also causes all dependent intermediate tables (and their descendants) to become unusable with a status of `BASE_TABLE_REMOVED`. The stored data in those intermediate tables is removed and storage-based billing stops. For more information, see Deleting an intermediate table.

To delete an ID mapping table

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Entity resolution** tab.

5. For ID mapping tables, choose a table.
6. On the ID mapping table details page, scroll down to view the **ID mapping tables**.
7. Choose and ID mapping table, and then choose **Delete**.
8. If you're certain that you want to delete the ID mapping table, choose **Delete**.

Analysis templates in AWS Clean Rooms

Analysis templates work with the Custom analysis rule in AWS Clean Rooms. With an analysis template, you can define parameters to help you reuse the same query. AWS Clean Rooms supports a subset of parameterization with literal values.

Analysis templates are collaboration-specific. For each collaboration, members can only see the queries in that collaboration. If you plan to use differential privacy in a collaboration, you should make sure that your analysis templates are compatible with the general-purpose query structure of AWS Clean Rooms Differential Privacy.

You can create an analysis template in two ways: using SQL code or using Python code for Spark.

Topics

- SQL analysis templates
- PySpark analysis templates
- Troubleshooting PySpark analysis templates

SQL analysis templates

SQL analysis templates enable you to query and analyze data across different datasets within a collaboration. You can use these templates to perform various types of analysis, such as identifying audience overlaps and calculating aggregated metrics.

With SQL analysis templates, you can:

- Write standard SQL queries
- Add parameters to make your queries dynamic
- Control access to specific columns and tables
- Set aggregation requirements for sensitive data
- Define input data for the generation of privacy-enhanced synthetic datasets for custom machine learning (ML) models

Topics

- Creating a SQL analysis template

- Reviewing a SQL analysis template

Creating a SQL analysis template

Prerequisites

Before you create a SQL analysis template, you must have:

- An active AWS Clean Rooms collaboration
- Access to at least one configured table in the collaboration

  For information about configuring tables in AWS Clean Rooms, see Creating a configured table in AWS Clean Rooms.
- Permissions to create analysis templates
- Basic knowledge of SQL query syntax

The following procedure describes the process of creating a SQL analysis template using the AWS Clean Rooms console.

For information about how to create a SQL analysis template using the AWS SDKs, see the AWS Clean Rooms API Reference.

To create a SQL analysis template

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Templates** tab, go to the **Analysis templates created by you** section.
5. Choose **Create analysis template**.
6. On the **Create analysis template** page, for **Details**,

   a. Enter a **Name** for the analysis template.

   b. (Optional) Enter a **Description**.

   c. For **Format**, leave the **SQL** option selected.
7. For **Tables**, view the configured tables associated with the collaboration.

8. For **Definition**,

   a. Enter the definition for the analysis template.

   b. Choose **Import from** to import a definition.

   c. (*Optional*) Specify a parameter in the SQL editor by entering a colon (`:`) in front of the parameter name.

      For example:

      ```
      WHERE table1.date + :date_period > table1.date
      ```

9. If you added parameters previously, under **Parameters – optional**, for each **Parameter name**, choose the **Type** and **Default value** (optional).

10. For **Synthetic data**, if you want to generate synthetic data for model training, select the **Require analysis template output to be synthetic** checkbox.

    For more information, see Privacy-enhanced synthetic dataset generation.

    a. For **Column classification**, choose a **Column** from the dropdown list. At least five columns are required.

       i. Choose a **Classification** from the dropdown list. This identifies the data type for each column.

          Classification types include:

          - **Numerical** – Continuous numerical values such as measurements or counts
          - **Categorical** – Discrete values or categories such as labels or types

       ii. To remove a column, select **Remove**.

       iii. To add another column, select **Add another column**. Choose the **Column** and **Classification** from the dropdown lists.

       iv. For **Predictive value**, choose a **Column** from the dropdown list. This is the column the custom model uses for prediction after it's trained on the synthetic dataset.

    b. **Advanced settings** allow you to set the **Privacy level** and **Privacy threshold**. Adjust the settings to fit your needs.

       - For **Privacy level**, enter an epsilon value to determine how much noise the synthetic model adds to protect privacy in your generated dataset. The value must be between 0.0001 and 10.

- Lower values add more noise, providing stronger privacy protection but potentially reducing utility for downstream custom model trained on this data.
- Higher values add less noise, providing more accuracy but potentially reducing privacy protection.

For **Privacy threshold**, enter the highest allowed probability that a membership inference attack could identify members of the original dataset. The value must be between 50.0 and 100.

- Scores of 50% indicate that a membership inference attack can't successfully distinguish members from non-members better than a random guess.
- For no privacy limit, enter 100%.

The optimal value depends on your specific use case and privacy requirements. If the privacy threshold is exceeded, the ML input channel creation fails, and you can't use the synthetic dataset to train a model.

> **Warning**
>
> Synthetic data generation protects against inferring individual attributes whether specific individuals are present in the original dataset or learning attributes of those individuals are present. However, it doesn't prevent literal values from the original dataset, including personally identifiable information (PII) from appearing in the synthetic dataset.
>
> We recommend avoiding values in the input dataset that are associated with only one data subject because these may re-identify a data subject. For example, if only one user lives in a zip code, the presence of that zip code in the synthetic dataset would confirm that user was in the original dataset. Techniques like truncating high precision values or replacing uncommon catalogues with *other* can be used to mitigate this risk. These transformations can be part of the query used to create the ML input channel.

11. If you want to enable **Tags** for the resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
12. Choose **Create**.

13. You are now ready to inform your collaboration member that they can Review an analysis template. (Optional if you want to query your own data.)

Reviewing a SQL analysis template

After a collaboration member has created a SQLanalysis template, you can review and approve it. After the analysis template and approved, it can be used in a query in AWS Clean Rooms.

> **Note**
>
> When you bring your analysis code into a collaboration, be aware of the following:
>
> - AWS Clean Rooms does not validate or guarantee the behavior of the analysis code.
> - If you need to ensure certain behavior, review the code of your collaboration partner directly or work with a trusted third-party auditor to review it.
> - In the shared security model:
> - You (the customer) are responsible for the security of the code running in the environment.
> - AWS Clean Rooms is responsible for the security of the environment, ensuring that
> - only the approved code runs
> - only specified configured tables are accessible
> - the only output destination is the result receiver's S3 bucket.

To review a SQL analysis template using the AWS Clean Rooms console

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Templates** tab, go to the **Analysis templates created by other members** section.
5. Choose the analysis template that has the **Can run status** of **No requires your review**.
6. Choose **Review**.
7. Review the analysis rule **Overview**, **Definition**, and **Parameters** (if any).

8. Review the configured tables listed under **Tables referenced in definition**.

   The **Status** next to each table will read **Template not allowed**.

9. Choose a table.

If you	Then choose
Approve the analysis template	**Allow template on table**. Confirm your approval by choosing **Allow**.
Don't approve the analysis template	**Disallow**

You are now ready to query the configured table using a SQL analysis template. For more information, see Running SQL queries.

PySpark analysis templates

PySpark analysis templates require a Python user script and an optional virtual environment to use custom and open-source libraries. These files are called artifacts.

Before you create an analysis template, you first create the artifacts and then store the artifacts in an Amazon S3 bucket. AWS Clean Rooms uses these artifacts when running analysis jobs. AWS Clean Rooms only accesses the artifacts when running a job.

Before running any code on a PySpark analysis template, AWS Clean Rooms validates artifacts by:

- Checking the specific S3 object version used when creating the template
- Verifying the SHA-256 hash of the artifact
- Failing any job where artifacts have been modified or removed

> **Note**
>
> The maximum size of all combined artifacts for a given PySpark analysis template in AWS Clean Rooms is 1 GB.

Security for PySpark analysis templates

To preserve a secure compute environment, AWS Clean Rooms uses a two-tier compute architecture to isolate user code from system operations. This architecture is based on Amazon EMR Serverless Fine Grained Access Control technology, also know as Membrane. For more information, see Membrane – Safe and performant data access controls in Apache Spark in the presence of imperative code.

The compute environment components are divided into a separate user space and system space. The user space executes the PySpark code in the PySpark analysis template. AWS Clean Rooms uses the system space to enable the job to run including using service roles provided by customers to read data to run the job and implementing the column allowlist. As a result of this architecture, a customer's PySpark code that affects the system space, which could include small number of Spark SQL and PySpark DataFrames APIs, is blocked.

PySpark limitations in AWS Clean Rooms

When customers submit an approved PySpark analysis template, AWS Clean Rooms runs it on its own secure compute environment which no customer can access. The compute environment implements a compute architecture with a user space and system space to preserve a secure compute environment. For more information, see Security for PySpark analysis templates.

Consider the following limitations before you use PySpark in AWS Clean Rooms.

Limitations

- Only DataFrame outputs are supported
- Single Spark session per job execution

Unsupported features

- **Data management**
  - Iceberg table formats
  - LakeFormation managed tables
  - Resilient distributed datasets (RDD)
  - Spark streaming
  - Access control for nested columns

- **Custom functions and extensions**
  - User-defined table functions (UDTFs)
  - HiveUDFs
  - Custom classes in user-defined functions
  - Custom data sources
  - Additional JAR files for:
    - Spark extensions
    - Connectors
    - Metastore configurations
- **Monitoring and analysis**
  - Spark logging
  - Spark UI
  - `ANALYZE TABLE` commands

> **Important**
>
> These limitations are in place to maintain the security isolation between user and system spaces.
> All restrictions apply regardless of collaboration configuration.
> Future updates may add support for additional features based on security evaluations.

Best practices

We recommend the following best practices when creating PySpark analysis templates.

- Design your analysis templates with the PySpark limitations in AWS Clean Rooms in mind.
- Test your code in a development environment first.
- Use supported DataFrame operations exclusively.
- Plan your output structure to work with DataFrame limitations.

We recommend the following best practices for managing artifacts

- Keep all PySpark analysis template artifacts in a dedicated S3 bucket or prefix.

- Use clear version naming for different artifact versions.
- Create new analysis templates when artifact updates are needed.
- Maintain an inventory of which templates use which artifact versions.

For more information about how to write Spark code, see the following:

- Apache Spark Examples
- Write a Spark application in the *Amazon EMR Release Guide*
- Tutorial: Writing an AWS Glue for Spark script in the *AWS Glue User Guide*

The following topics explain how to create Python user scripts and libraries before creating and reviewing the analysis template.

Topics

- Creating a user script
- Working with parameters in PySpark analysis templates
- Creating a virtual environment (optional)
- Storing a user script and virtual environment in S3
- Creating a PySpark analysis template
- Reviewing a PySpark analysis template

Creating a user script

The user script must contain an entrypoint function (in other words, a handler). You can name your user script file with any valid Python filename.

The following procedure describes how to create a user script to define the core functionality of your PySpark analysis.

Prerequisites

- PySpark 1.0 (corresponds to Python 3.11 and Spark 3.5.3)
- Datasets in Amazon S3 can only be read as configured table associations in the Spark session you define.
- Your code can't directly call Amazon S3 and AWS Glue

- Your code can't make network calls

To create a user script

1. Open a text editor or Integrated Development Environment (IDE) of your choice.

   You can use any text editor or IDE (such as Visual Studio Code, PyCharm, or Notepad++) that supports Python files.

2. Create a new Python file with a name of your choice (for example, **`my_analysis.py`**).
3. Define an entrypoint function that accepts a context object parameter.

   ```
   def entrypoint(context)
   ```

   The `context` object parameter is a dictionary that provides access to essential Spark components, referenced tables, and analysis parameters. It contains:

   Spark session access via `context['sparkSession']`

   Referenced tables via `context['referencedTables']`

   Analysis parameters via `context['analysisParameters']` (if parameters are defined in the template)

4. Define the results of the entrypoint function:

   ```
   return results
   ```

   The `results` must return an object containing a results dictionary of filenames to an output DataFrame.

   > **Note**
   >
   > AWS Clean Rooms automatically writes the DataFrame objects to the S3 bucket of the result receiver.

5. You are now ready to:

   a. Store this user script in S3. For more information, see Storing a user script and virtual environment in S3.

b. Create the optional virtual environment to support any additional libraries required by your user script. For more information, see Creating a virtual environment (optional).

Example 1

<caption>The following example demonstrates a generic user script for a PySpark analysis template.</caption>

```
# File name: my_analysis.py

def entrypoint(context):
    try:
        # Access Spark session
        spark = context['sparkSession']

        # Access input tables
        input_table1 = context['referencedTables']['table1_name']
        input_table2 = context['referencedTables']['table2_name']

        # Example data processing operations
        output_df1 = input_table1.select("column1", "column2")
        output_df2 = input_table2.join(input_table1, "join_key")
        output_df3 = input_table1.groupBy("category").count()

        # Return results - each key creates a separate output folder
        return {
            "results": {
                "output1": output_df1,        # Creates output1/ folder
                "output2": output_df2,        # Creates output2/ folder
                "analysis_summary": output_df3 # Creates analysis_summary/ folder
            }
        }

    except Exception as e:
        print(f"Error in main function: {str(e)}")
        raise e
```

The folder structure of this example is as follows:

```
analysis_results/
#
### output1/ # Basic selected columns
```

```
# ### part-00000.parquet
# ### _SUCCESS
#
### output2/ # Joined data
# ### part-00000.parquet
# ### _SUCCESS
#
### analysis_summary/ # Aggregated results
### part-00000.parquet
### _SUCCESS
```

Example 2

<caption>The following example demonstrates a more complex user script for a PySpark analysis template.</caption>

```
def entrypoint(context):
    try:
        # Get DataFrames from context
        emp_df = context['referencedTables']['employees']
        dept_df = context['referencedTables']['departments']

        # Apply Transformations
        emp_dept_df = emp_df.join(
            dept_df,
            on="dept_id",
            how="left"
        ).select(
            "emp_id",
            "name",
            "salary",
            "dept_name"
        )

        # Return Dataframes
        return {
            "results": {
                "outputTable": emp_dept_df
            }
        }

    except Exception as e:
        print(f"Error in entrypoint function: {str(e)}")
```

```
        raise e
```

Working with parameters in PySpark analysis templates

Parameters increase the flexibility of your PySpark analysis templates by allowing different values to be provided at job submission time. Parameters are accessible through the context object passed to your entrypoint function.

> **Note**
>
> Parameters are user-provided strings that can contain arbitrary content.
>
> - Review the code to ensure parameters are handled safely to prevent unexpected behavior in your analysis.
> - Design parameter handling to work safely regardless of what parameter values are provided at submission time.

Accessing parameters

Parameters are available in the `context['analysisParameters']` dictionary. All parameter values are strings.

Example Accessing parameters safely

```
def entrypoint(context):
    # Access parameters from context
    parameters = context['analysisParameters']
    threshold = parameters['threshold']
    table_name = parameters['table_name']

    # Continue with analysis using parameters
    spark = context['sparkSession']
    input_df = context['referencedTables'][table_name]

    # Convert threshold value
    threshold_val = int(threshold)

    # Use parameter in DataFrame operation
    filtered_df = input_df.filter(input_df.amount > threshold_val)
```

```
    return {
        "results": {
            "output": filtered_df
        }
    }
```

Parameter security best practices

> **Warning**
>
> Parameters are user-provided strings that can contain arbitrary content. You must handle parameters safely to prevent security vulnerabilities in your analysis code.

Unsafe parameter handling patterns to avoid:

- **Executing parameters as code** – Never use `eval()` or `exec()` on parameter values

```
# UNSAFE - Don't do this
eval(parameters['expression'])  # Can execute arbitrary code
```

- **SQL string interpolation** – Never concatenate parameters directly into SQL strings

```
# UNSAFE - Don't do this
sql = f"SELECT * FROM table WHERE column = '{parameters['value']}'"  # SQL injection
 risk
```

- **Unsafe file path operations** – Never use parameters directly in file system operations without validation

```
# UNSAFE - Don't do this
file_path = f"/data/{parameters['filename']}"  # Path traversal risk
```

Safe parameter handling patterns:

- **Use parameters in DataFrame operations** – Spark DataFrames handle parameter values safely

```
# SAFE - Use parameters in DataFrame operations
threshold = int(parameters['threshold'])
```

```
filtered_df = input_df.filter(input_df.value > threshold)
```

- **Validate parameter values** – Check that parameters meet expected formats before use

```
# SAFE - Validate parameters before use
def validate_date(date_str):
    try:
        from datetime import datetime
        datetime.strptime(date_str, '%Y-%m-%d')
        return True
    except ValueError:
        return False

date_param = parameters['date_filter'] or '2024-01-01'
if not validate_date(date_param):
    raise ValueError(f"Invalid date format: {date_param}")
```

- **Use allowlists for parameter values** – When possible, validate parameters against known good values

```
# SAFE - Use allowlists
allowed_columns = ['column1', 'column2', 'column3']
column_param = parameters['column_name']
if column_param not in allowed_columns:
    raise ValueError(f"Invalid column: {column_param}")
```

- **Type conversion with error handling** – Convert string parameters to expected types safely

```
# SAFE - Convert with error handling
try:
    batch_size = int(parameters['batch_size'] or '1000')
    if batch_size <= 0 or batch_size > 10000:
        raise ValueError(f"Batch size must be between 1 and 10000")
except ValueError as e:
    print(f"Invalid parameter: {e}")
    raise
```

> **Important**
>
> Remember that parameters bypass code review when job runners provide different values. Design your parameter handling to work safely regardless of what parameter values are provided.

Complete parameter example

Example Using parameters safely in a PySpark script

```
def entrypoint(context):
    try:
        # Access Spark session and tables
        spark = context['sparkSession']
        input_table = context['referencedTables']['sales_data']

        # Access parameters - fail fast if analysisParameters missing
        parameters = context['analysisParameters']

        # Validate and convert numeric parameter (handles empty strings with default)
        try:
            threshold = int(parameters['threshold'] or '100')
            if threshold <= 0:
                raise ValueError("Threshold must be positive")
        except (ValueError, TypeError) as e:
            print(f"Invalid threshold parameter: {e}")
            raise

        # Validate date parameter (handles empty strings with default)
        date_filter = parameters['start_date'] or '2024-01-01'
        from datetime import datetime
        try:
            datetime.strptime(date_filter, '%Y-%m-%d')
        except ValueError:
            raise ValueError(f"Invalid date format: {date_filter}")

        # Use parameters safely in DataFrame operations
        filtered_df = input_table.filter(
            (input_table.amount > threshold) &
            (input_table.date >= date_filter)
        )
```

```
        result_df = filtered_df.groupBy("category").agg(
            {"amount": "sum"}
        )

        return {
            "results": {
                "filtered_results": result_df
            }
        }

    except Exception as e:
        print(f"Error in analysis: {str(e)}")
        raise
```

Creating a virtual environment (optional)

If you have any additional libraries required by your user script, you have the option to create a virtual environment to store those libraries. If you don't need additional libraries, you can skip this step.

When working with libraries that have native extensions, keep in mind that PySpark in AWS Clean Rooms operates on Linux with ARM64 architecture.

The following procedure demonstrates how to create a virtual environment using a basic CLI command.

To create a virtual environment

1. Open a terminal or command prompt.
2. Add the following content:

```
# create and activate a python virtual environment
python3 -m venv pyspark_venvsource
source pyspark_venvsource/bin/activate

# install the python packages
pip3 install pandas # add packages here

# package the virtual environment into an archive
pip3 install venv-pack
venv-pack -f -o pyspark_venv.tar.gz
```

```
# optionally, remove the virtual environment directory
deactivate
rm -fr pyspark_venvsource
```

3. You are now ready to store this virtual environment in S3. For more information, see Storing a user script and virtual environment in S3.

For more information about working with Docker and Amazon ECR, see the Amazon ECRUser Guide.

Storing a user script and virtual environment in S3

The following procedure explains how to store a user script and optional virtual environment in Amazon S3. Complete this step before creating a PySpark analysis template.

> **Important**
>
> Do not modify or remove artifacts (user scripts or virtual environments) after creating an analysis template.
> Doing so will:
>
> - Cause all future analysis jobs using this template to fail.
> - Require creation of a new analysis template with new artifacts.
> - Not affect previously completed analysis jobs

Prerequisites

- An AWS account with appropriate permissions
- A user script file (such as `my_analysis.py`)
- (Optional, if one exists) A virtual environment package (`.tar.gz` file)
- Access to create or modify IAM roles

Console

To store a user script and virtual environment in S3 using the console:

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Create a new S3 bucket or use an existing one.
3. Enable versioning for the bucket.
   a. Select your bucket.
   b. Choose **Properties**.
   c. In the **Bucket Versioning** section, choose **Edit**.
   d. Select **Enable** and save changes.
4. Upload your artifacts and enable SHA-256 hash.
   a. Navigate to your bucket.
   b. Choose **Upload**.
   c. Choose **Add files** and add your user script file.
   d. (Optional, if one exists) Add your **.tar.gz** file.
   e. Expand **Properties**.
   f. Under **Checksums**, for **Checksum function**, select **SHA256**.
   g. Choose **Upload**.
5. You are now ready to create a PySpark analysis template.

CLI

To store the user script and virtual environment in S3 using the AWS CLI:

1. Run the following command:

```
aws s3 cp --checksum-algorithm sha256 pyspark_venv.tar.gz s3://ARTIFACT-BUCKET/
EXAMPLE-PREFIX/
```

2. You are now ready to create a PySpark analysis template.

> **Note**
>
> If you need to update script or virtual environment:
>
> 1. Upload the new version as a separate object.
> 2. Create a new analysis template using the new artifacts.
> 3. Deprecate the old template.
> 4. Keep the original artifacts in S3 if the old template might still be needed.

Creating a PySpark analysis template

> **Note**
>
> Parameters are user-provided strings that can contain arbitrary content.
>
> - Review the code to ensure parameters are handled safely to prevent unexpected behavior in your analysis.
> - Design parameter handling to work safely regardless of what parameter values are provided at submission time.

Prerequisites

Before you create a PySpark analysis template, you must have:

- A membership in an active AWS Clean Rooms collaboration
- Access to at least one configured table in the active collaboration
- Permissions to create analysis templates
- A Python user script and a virtual environment created and stored in S3
  - S3 bucket has versioning enabled. For more information, see Using versioning in S3 buckets
  - S3 bucket can calculate SHA-256 checksums for uploaded artifacts. For more information, see Using checksums
- Permissions to read code from an S3 bucket

  For information about creating the required service role, see Create a service role to read code from an S3 bucket (PySpark analysis template role).

The following procedure describes the process of creating a PySpark analysis template using the AWS Clean Rooms console. It assumes that you have already created a user script and virtual environment files and stored your user script and virtual environment files in an Amazon S3 bucket.

> **Note**
>
> The member who creates the PySpark analysis template must also be the member who receives results.

For information about how to create a PySpark analysis template using the AWS SDKs, see the AWS Clean Rooms API Reference.

To create a PySpark analysis template

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. On the **Templates** tab, go to the **Analysis templates created by you** section.
5. Choose **Create analysis template**.
6. On the **Create analysis template** page, for **Details**,
   a. Enter a **Name** for the analysis template.
   b. (Optional) Enter a **Description**.
   c. For **Format**, choose the **PySpark** option.
7. For **Definition**,
   a. Review the **Prerequisites** and ensure each prerequisite is met before continuing.
   b. For **Entry point file**, enter the S3 bucket or choose **Browse S3**.
   c. (Optional) For **Libraries file**, enter the S3 bucket or choose **Browse S3**.
8. For **Parameters – optional**, if you want to add parameters to make your analysis template reusable:
   a. Choose **Add parameter**.

b. Enter a **Parameter name**.

   Parameter names must start with a letter or underscore, followed by alphanumeric characters or underscores.

c. For **Type**, **STRING** is automatically selected as the only supported type for PySpark analysis templates.

d. (Optional) Enter a **Default value** for the parameter.

   If you provide a default value, job runners can use this value when running jobs without explicitly providing a parameter value.

e. To add more parameters, choose **Add another parameter** and repeat the previous steps.

> **Note**
>
> You can define up to 50 parameters per PySpark analysis template. Each parameter value can be up to 1,000 characters.

9. For **Tables referenced in the definition**,

   - If all tables referenced in the definition have been associated to the collaboration:
     - Leave the **All tables referenced in the definition have been associated to the collaboration** checkbox selected.
     - Under **Tables associated to the collaboration**, choose all associated tables that are referenced in the definition.
   - If all tables referenced in the definition haven't been associated to the collaboration:
     - Clear the **All tables referenced in the definition have been associated to the collaboration** checkbox.
     - Under **Tables associated to the collaboration**, choose all associated tables that are referenced in the definition.
     - Under **Tables that will be associated later**, enter a table name.
     - Choose **List another table** to list another table.

10. For **Error message configuration**, choose one of the following:

    - **Basic error messages** – returns basic error messages without exposing underlying data. Recommended for production workloads.

- **Detailed error messages** – returns detailed error messages for faster troubleshooting. Recommended in development and testing environments. May expose sensitive data, including personally identifiable information (PII).

> **Note**
>
> When using **Detailed error messages**, all data provider members must approve this setting for the template.

11. Specify the **Service access** permissions by selecting an **Existing service role name** from the dropdown list.

    1. The list of roles are displayed if you have permissions to list roles.

       If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.

    2. View the service role by choosing the **View in IAM** external link.

       If there are no existing service roles, the option to **Use an existing service role** is unavailable.

       By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

> **Note**
>
> - AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
> - If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
> - If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

12. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
13. Choose **Create**.
14. You are now ready to inform your collaboration member that they can Review an analysis template. (Optional if you want to query your own data.)

> **Important**
>
> Don't modify or remove artifacts (user scripts or virtual environments) after creating an analysis template.
> Doing so will:
>
> - Cause all future analysis jobs using this template to fail.
> - Require creation of a new analysis template with new artifacts.
> - Not affect previously completed analysis jobs.

Reviewing a PySpark analysis template

When another member creates an analysis template in your collaboration, you must review and approve it before it can be used.

The following procedure shows you how to review a PySpark analysis template, including its rules, parameters, and referenced tables. As a collaboration member, you'll assess whether the template aligns with your data sharing agreements and security requirements.

After the analysis template and approved, it can be used in a job in AWS Clean Rooms.

> **Note**
>
> When you bring your analysis code into a collaboration, be aware of the following:
>
> - AWS Clean Rooms doesn't validate or guarantee the behavior of the analysis code.
> - If you need to ensure certain behavior, review the code of your collaboration partner directly or work with a trusted third-party auditor to review it.
> - AWS Clean Rooms guarantees that the SHA-256 hashes of the code listed in the PySpark analysis template matches the code running in the PySpark analysis environment.

- AWS Clean Rooms doesn't perform any auditing or security analysis of additional libraries you bring into the environment.
- In the shared security model:
  - You (the customer) are responsible for the security of the code running in the environment.
  - You (the customer) are responsible for setting the appropriate error message configuration for the environment.
  - AWS Clean Rooms is responsible for the security of the environment, ensuring that
    - only the approved code runs
    - only specified configured tables are accessible
    - the only output destination is the result receiver's S3 bucket.

AWS Clean Rooms generates SHA-256 hashes of the user script and virtual environment for your review. However, the actual user script and libraries aren't directly accessible within AWS Clean Rooms.

To validate that the user script and libraries shared are the same as those referenced in the analysis template, you can create a SHA-256 hash of the files shared and compare it to the analysis template hash created by AWS Clean Rooms. The hashes of the code run will also be in the job logs.

Prerequisites

- Linux/Unix operating system or Windows Subsystem for Linux (WSL)
- User script file you want to hash
  - Request that the analysis template creator share the file through a secure channel.
- The analysis template hash created by AWS Clean Rooms

To review a PySpark analysis template using the AWS Clean Rooms console

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with the AWS account that will function as the collaboration creator.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.

4. On the **Templates** tab, go to the **Analysis templates created by other members** section.
5. Choose the analysis template that has the **Can run status** of **No requires your review**.
6. Choose **Review**.
7. Review the analysis rule **Overview**, **Definition**, and **Parameters** (if any).

> **Note**
>
> Parameters allow analysis runners to submit different values at submission time. If an analysis template supports parameters, review how the parameter values are used in the code of your collaboration partner to ensure it meets your requirements.

8. Validate that the shared user script and libraries are the same as those referenced in the analysis template.

   a. Create a SHA-256 hash of the files shared and compare it to the analysis template hash created by AWS Clean Rooms.

      You can generate a hash by navigating to the directory containing your user script file and then running the following command:

      ```
      sha256sum your_script_filename.py
      ```

      Example output:

      ```
      e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 my_analysis.py
      ```

   b. Alternatively, you can use Amazon S3 checksum features. For more information, see Checking object integrity in Amazon S3 in the *Amazon S3 User Guide*.

   c. Another alternative is to view the hashes of the executed code in the job logs.

9. Review the configured tables listed under **Tables referenced in definition**.

   The **Status** next to each table will read **Template not allowed**.

10. Choose a table.

    a. To approve the analysis template, choose **Allow template on table**. Confirm your approval by choosing **Allow**.

    b. To decline approval, choose **Disallow**.

If you have chosen to approve the analysis template, the member who can run jobs can now run a PySpark job on a configured table using a PySpark analysis template. For more information, see Running PySpark jobs.

Troubleshooting PySpark analysis templates

When running jobs using PySpark analysis templates, you might encounter failures during job initialization or execution. These failures typically relate to script configuration, data access permissions, or environment setup.

For more information about PySpark limitations, see PySpark limitations in AWS Clean Rooms.

Topics

- Troubleshooting your code
- Analysis template job doesn't start
- Analysis template job starts but fails during processing
- Virtual environment setup fails

Troubleshooting your code

To help you develop and troubleshoot your code, we suggest you simulate AWS Clean Rooms in your own AWS account by enabling **Detailed error messages** and running jobs using your own test data.

You can also simulate PySpark in AWS Clean Rooms in Amazon EMR Serverless with the following steps. It will have small differences with PySpark in AWS Clean Rooms but mostly covers how your code can be run.

To simulate PySpark in AWS Clean Rooms in EMR Serverless

1. Create a dataset in Amazon S3, catalog it in the AWS Glue Data Catalog, and set up Lake Formation permissions.
2. Register the S3 location with Lake Formation using a custom role.
3. Create an Amazon EMR Studio instance if you don’t already have one (Amazon EMR Studio is needed to use Amazon EMR Serverless).

4. Create an EMR Serverless app
   - Select release version emr-7.7.0.
   - Select ARM64 architecture.
   - Opt for **Use custom settings**.
   - Disable preinitialized capacity.
   - If you plan to do interactive work, select **Interactive endpoint** > **Enable endpoint for EMR studio**.
   - Select **Additional configurations** > **Use Lake Formation for fine-grained access control**.
   - Create the application.
5. Use EMR-S either via EMR-Studio notebooks or the `StartJobRun` API.

Analysis template job doesn't start

Common causes

Analysis template jobs can fail immediately at startup due to three main configuration issues:

- Incorrect script naming that doesn't match the required format
- Missing or incorrectly formatted entrypoint function in the user script
- Incompatible Python version in the virtual environment

Resolution

To resolve:

1. Verify your user script:
   - Check that your user script has a valid Python filename.

     Valid Python filenames use lowercase letters, underscores to separate words, and the .py extension.
2. Verify the entrypoint function. If your user script doesn't have an entrypoint function, add one.
   a. Open your user script.
   b. Add this entrypoint function:

```
def entrypoint(context):
    # Your analysis code here
```

c. Ensure the function name is spelled exactly as `entrypoint`.

d. Verify the function accepts the `context` parameter.

3. Check Python version compatibility:

   a. Verify your virtual environment uses Python 3.9 or 3.11.

   b. To check your version, run: `python --version`

   c. If needed, update your virtual environment:

```
conda create -n analysis-env python=3.9
conda activate analysis-env
```

Prevention

- Use the provided analysis template starter code that includes the correct file structure.
- Set up a dedicated virtual environment with Python 3.9 or 3.11 for all analysis templates.
- Test your analysis template locally using the template validation tool before submitting jobs.
- Implement CI/CD checks to verify script naming and entrypoint function requirements.

Analysis template job starts but fails during processing

Common causes

Analysis jobs can fail during execution for these security and formatting reasons:

- Unauthorized direct access attempts to AWS services like Amazon S3 or AWS Glue
- Returning output in incorrect formats that don't match required DataFrame specifications
- Blocked network calls due to security restrictions in the execution environment

Resolution

To resolve

1. Remove direct AWS service access:
   a. Search your code for direct AWS service imports and calls.
   b. Replace direct S3 access with provided Spark session methods.
   c. Use only pre-configured tables through the collaboration interface.
2. Format outputs correctly:
   a. Verify all outputs are Spark DataFrames.
   b. Update your return statement to match this format:

```
return {
    "results": {
        "output1": dataframe1
    }
}
```

   c. Remove any non-DataFrame return objects.
3. Remove network calls:
   a. Identify and remove any external API calls.
   b. Remove any urllib, requests, or similar network libraries.
   c. Remove any socket connections or HTTP client code.

Prevention

- Use the provided code linter to check for unauthorized AWS imports and network calls.
- Test jobs in the development environment where security restrictions match production.
- Follow the output schema validation process before deploying jobs.
- Review the security guidelines for approved service access patterns.

Virtual environment setup fails

Common causes

Virtual environment configuration failures commonly occur due to:

- Mismatched CPU architecture between development and execution environments
- Python code formatting issues that prevent proper environment initialization
- Incorrect base image configuration in container settings

Resolution

To resolve

1. Configure the correct architecture:
   a. Check your current architecture with `uname -m`.
   b. Update your Dockerfile to specify ARM64:

   ```
   FROM --platform=linux/arm64 public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
   ```

   c. Rebuild your container with `docker build --platform=linux/arm64`.
2. Fix Python indentation:
   a. Run a Python code formatter like `black` on your code files.
   b. Verify consistent use of spaces or tabs (not both).
   c. Check indentation of all code blocks:

   ```
   def my_function():
       if condition:
           do_something()
       return result
   ```

   d. Use an IDE with Python indentation highlighting.
3. Validate environment configuration:
   a. Run `python -m py_compile your_script.py` to check for syntax errors.
   b. Test the environment locally before deployment.

c. Verify all dependencies are listed in `requirements.txt`.

Prevention

- Use Visual Studio Code or PyCharm with Python formatting plugins
- Configure pre-commit hooks to run code formatters automatically
- Build and test environments locally using the provided ARM64 base image
- Implement automated code style checking in your CI/CD pipeline

Intermediate tables in AWS Clean Rooms

An *intermediate table* is a collaboration-scoped resource in AWS Clean Rooms that caches analysis results within a collaboration for reuse in subsequent analyses.

Intermediate tables are created by the analysis runner who owns and manages them. The data is stored in AWS managed storage and is not exported outside the collaboration.

Before you can use an intermediate table in an analysis, you must first populate it. Then, attach a Custom analysis rule to it. The table must have a status of `POPULATE_SUCCESS` before it can be referenced in any analysis. You can reference intermediate tables by name in SQL queries and PySpark jobs, just like configured tables.

> **Note**
>
> Storage for intermediate tables is billed at standard rates until the table is deleted.

A *base table* is any configured table, ID mapping table, or other intermediate table that is used in the analysis to populate an intermediate table.

Deferred enforcement

AWS Clean Rooms uses *deferred enforcement* for certain privacy controls on intermediate tables. AWS Clean Rooms does not evaluate deferred controls when you create or populate the intermediate table. Instead, AWS Clean Rooms enforces them at analysis time, when the intermediate table is referenced in a subsequent analysis.

Intermediate tables have the following classification types, which determine how deferred controls are applied:

- **First-party** – All base tables used to populate the intermediate table are owned by the creator. The creator has full control over the analysis rule configuration.
- **Multi-party** – Base tables come from different data providers. The creator can only make inherited controls more restrictive, not less.

The intermediate table inherits deferred controls from the base tables used to populate it. When you populate an intermediate table, AWS Clean Rooms captures the deferred controls from each

base table's analysis rule. It then applies the most restrictive combination to the intermediate table.

The following table describes the deferred controls for intermediate tables.

Deferred control	Description
Allowed result receivers	Specifies which collaboration members can receive the results of an analysis that references the intermediate table. Inherited as the intersection of all base tables' allowed result receivers.
Disallowed output columns	Specifies columns that cannot appear in the output of an analysis. Inherited as the union of all base tables' disallowed output columns.
Allowed additional analyses	Specifies whether the results of an analysis can be used as input to additional analyses. Inherited as the most restrictive value from the base tables.
Additional analyses	Controls whether the intermediate table can be used in additional analyses beyond the initial analysis. Inherited as the most restrictive value from the base tables.

Intermediate table statuses

The following table describes the status lifecycle for an intermediate table.

Status	Description
`CREATED`	The intermediate table resource exists but has no materialized data.
`POPULATE_STARTED`	A populate operation is in progress.
`POPULATE_SUCCESS`	Data has been successfully materialized in the intermediate table.
`POPULATE_FAILED`	The populate operation did not complete successfully.

Status	Description
`DISALLOWED_BY_DATA_PROVIDER`	A data provider has disallowed the intermediate table. The table can no longer be used in analyses until the data provider re-allows it.
`BASE_TABLE_REMOVED`	One or more base tables used to populate the intermediate table have been removed from the collaboration. The intermediate table can no longer be used in analyses.
`RETENTION_PERIOD_EXPIRED`	The intermediate table version has expired based on the configured retention period. All stored data has been cleaned up and the table must be repopulated in order to be used again. The default retention is 30 days.

Budgets with intermediate tables

You can configure access budgets and differential privacy budgets on intermediate tables, the same as configured tables.

Access budgets

AWS Clean Rooms decrements access budgets for the intermediate table and all base tables referenced in the analysis (including transitive dependencies for nested intermediate tables) at the following times:

- **Creation or refresh** – Budgets are decremented because the populate operation executes an analysis against base tables.
- **Usage** – Budgets are decremented because the intermediate table contains base table data. Every access to that data, even if cached, counts as a use of the base table for privacy tracking.

Differential privacy budgets

AWS Clean Rooms handles differential privacy budgets for intermediate tables as follows:

- **Creation or refresh** – If base tables have differential privacy enabled, their differential privacy budgets are decremented. AWS Clean Rooms injects noise at this time. The analysis that

populates the intermediate table must follow all restrictions for queries with differential privacy enabled.

- **Usage** – If you configured differential privacy for the intermediate table, the epsilon for the collaboration's single budget owner is decremented. This applies regardless of who owns the intermediate table. If no budget exists, the analysis is rejected.

> **Note**
>
> Differential privacy on intermediate tables treats the data in an intermediate table as a fresh dataset. AWS Clean Rooms injects noise based on the number of users in the intermediate table itself. If you amplify the number of users through a transformation when populating the intermediate table, the effect of that transformation on the original user population will not be tracked.

Topics

- Creating an intermediate table
- Populating an intermediate table
- Configuring an analysis rule for an intermediate table
- Deleting an intermediate table

Creating an intermediate table

You can create an intermediate table to cache analysis results for reuse in subsequent queries within a collaboration.

To create an intermediate table

1. Open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration.
4. Choose the **Tables** tab.
5. From the **Add** dropdown, choose **Create intermediate table**.
6. In **Details**, enter a **Name** for the intermediate table and an optional **Description**.

> **Note**
>
> The name must be unique across all tables, intermediate tables, and ID mapping tables in the collaboration. The name can be up to 100 characters and cannot contain SQL reserved words. The description can be up to 255 characters.

7. In **Details**, choose a **Retention (in days)**. The default retention period is 30 days. This period applies to any active intermediate table version. After a version expires, all stored data is deleted from AWS Clean Rooms. To use the table again, you must repopulate it.
8. In **Dataset**, choose a method to define your intermediate table:
   - **SQL query** – Use the Tables panel to reference available tables, and write a query in the SQL editor.
   - **Analysis template** – Provide the ARN of a PySpark analysis template.
9. In **Encryption**, optionally select **Encrypt secret with a custom KMS key** and provide the KMS key ARN. For more information about the required key policy, see Encryption for intermediate tables.
10. (Optional) Add **Tags** to organize and identify your intermediate table.
11. In **Populate**, optionally select **Populate the intermediate table after creation**.

    If you select this option, configure the following:

    - **Query compute payer** – If your collaboration has multiple query compute payers, select the one that pays for query compute costs for the populate operation.
    - **Worker type** – The instance type for the populate job (default: CR.1X).
    - **Number of workers** – The number of instances to use (2–128, default: 16).
    - (Optional) **Spark properties** – Custom Spark runtime configuration.
12. Choose **Create intermediate table**.

> **Note**
>
> The service creates the intermediate table in CREATED status. The table contains no materialized data until you populate it. If you didn't select the option to populate after

creation, you can populate the table later. For more information, see Populating an intermediate table.

Populating an intermediate table

After you create an intermediate table, you must populate it to materialize data. You can also repopulate an intermediate table to refresh the data when base tables are updated.

Each populate operation creates a new version of the intermediate table. The previous version remains available during the refresh. If the refresh fails, the existing data is not affected.

> **Note**
>
> Populating an intermediate table decrements access budgets on all referenced base tables, including transitive dependencies.

> **Note**
>
> Populating an intermediate table starts a query to store the data in the service. This query counts towards your quotas for concurrent SQL queries per account and concurrent SQL query vCPU usage per account. For more information, see AWS Clean Rooms quotas.

To populate an intermediate table

1. Open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration, and then choose the **Tables** tab.
4. Choose the intermediate table that you want to populate.
5. Choose **Populate**.
6. In the **Populate** dialog, review the stored analysis (SQL query or analysis template reference).
7. Configure the following settings:

   - **Query compute payer** – If your collaboration has multiple query compute payers, select the one that pays for query compute costs for the populate operation.

- **Worker type** – The instance type for the populate job (default: CR.1X).
- **Number of workers** – The number of instances to use (2–128, default: 16).
- (Optional) **Spark properties** – Custom Spark runtime configuration.

8. Choose **Populate**.

Tracking progress

To view the status of a populate operation, choose the **Analysis** tab on the intermediate table details page. You can view the following information:

- Protected query ID (linked to query details)
- Status
- Worker type
- Number of workers
- Billed CRPU hours

Configuring an analysis rule for an intermediate table

An intermediate table is not available for analysis until you attach an analysis rule to it. The only supported analysis rule type for intermediate tables is Custom.

You can view the inherited constraints from base tables on the **Analysis rule** tab of the intermediate table details page. Each constraint displays a "View sources" link that shows which data provider's table imposed the value.

For first-party intermediate tables (all base tables owned by the creator), you have full control over the analysis rule configuration. For multi-party intermediate tables (base tables from different providers), you can only make inherited controls more restrictive.

> **Note**
>
> AWS Clean Rooms generates inherited constraints as a point-in-time snapshot of the base table analysis rules when you populate an intermediate table. Inherited constraints do not get updated when a base table analysis rule changes asynchronously. To refresh inherited constraints, populate the intermediate table again.

The following table describes the validation rules for multi-party intermediate tables.

Deferred control	Validation
allowedResultReceivers	Must be a subset of the intersection of all base table values.
disallowedOutputColumns	Must be a superset of the union of all base table values.
allowedAdditionalAnalyses	Must be a subset of the intersection of all base table values.
additionalAnalyses	Must be equal to or more restrictive (REQUIRED > ALLOWED > NOT_ALLOWED).

To configure an analysis rule for an intermediate table

1. Open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration, and then choose the **Tables** tab.
4. Choose the intermediate table.
5. In the alert at the top of the intermediate table details page, choose **Configure analysis rule**.
6. In **Step 1 - Specify analysis controls**, choose the analysis control method:

   - **Review each new analysis** – Requires approval of each analysis template before use.
   - **Allow any analysis by specific collaborators** – Allows specified collaborators to run any analysis.

   (Optional) Add allowed analysis templates (up to 50).
7. In **Step 2 - Specify analysis results controls**, configure the following:

   - **Columns not allowed in output** – Choose **None** or **Custom list** and select the columns to disallow.
   - **Additional analyses applied to output** – Choose **Not allowed**, **Allowed**, or **Required**.
   - **Results delivery** – Select the members who can receive results.
8. In **Step 3 - Set differential privacy**, choose to turn differential privacy on or off.
9. In **Step 4 - Review and configure**, review all settings.

Use the **Edit** buttons to modify any settings. You can preview what members see by choosing **Table view** or **JSON**.

Review the **Columns available for analysis** table, which shows the column name, type, and whether each column is allowed in query output.

10. Choose **Configure analysis rule**.

Deleting an intermediate table

You can remove an intermediate table from a collaboration by deleting it as the owner. Other collaboration actions can cause an intermediate table to reach an unusable state without deleting the resource (such as `DISALLOWED_BY_DATA_PROVIDER` or `BASE_TABLE_REMOVED`). In any of these deletion scenarios, the stored data in AWS Clean Rooms is deleted, and storage-based billing stops.

To delete an intermediate table

1. Open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms/.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration, and then choose the **Tables** tab.
4. Choose the intermediate table that you want to delete.
5. Choose **Delete**.
6. To confirm deletion, choose **Delete**.

Other deletion triggers

In addition to explicit owner deletion, an intermediate table can become unusable through the following triggers which will delete the corresponding stored data in AWS Clean Rooms from the intermediate table.

Data provider disallowing

A data provider can disallow a specific intermediate table that references their base table. This deletes the corresponding stored data in AWS Clean Rooms from the intermediate table. The intermediate table status is changed to `DISALLOWED_BY_DATA_PROVIDER`. The data provider can optionally cascade the disallow action to descendant intermediate tables by using the

`includeDescendants` flag, which will delete the corresponding data stored in AWS Clean Rooms from descendant tables.

Cascading behavior

When base tables (configured table associations or ID mapping tables) are removed from a collaboration, dependent intermediate tables become unusable with a status of `BASE_TABLE_REMOVED` and the corresponding data stored in AWS Clean Rooms is deleted from the dependent intermediate tables. The owner must call `DeleteIntermediateTable` to remove the resource.

Analyze data in a collaboration

In AWS Clean Rooms, you can analyze data by running queries or jobs.

A *query* is a method to access and analyze configured tables in a collaboration, using a supported set of functions, classes, and variables. The currently supported query language in AWS Clean Rooms is SQL. There are 3 ways to run a query in AWS Clean Rooms: Write SQL code, use an approved SQL analysis template, or use the Analysis builder UI.

A *job* is a method to access and analyze configured tables in a collaboration using a supported set of functions, classes, and variables. The currently supported type of job in AWS Clean Rooms is PySpark. There is one way to run a job in AWS Clean Rooms: by using an approved PySpark analysis template.

The following topics describe how to analyze data in AWS Clean Rooms by running SQL queries or PySpark jobs.

Topics

- Running SQL queries
- Running PySpark jobs
- Using an intermediate table in analyses

Running SQL queries

> **Note**
>
> You can only run queries if the member who is responsible to pay for query compute costs has joined the collaboration as an active member.

As the member who can query, you can run a SQL query by:

- Building a SQL query manually using the SQL code editor.
- Using an approved SQL analysis template.
- Using the **Analysis builder UI** to build a query without having to write SQL code.

When the member who can query runs a SQL query on the tables in the collaboration, AWS Clean Rooms assumes the relevant roles to access the tables on their behalf. AWS Clean Rooms applies the analysis rules as necessary to the input query and its output.

The analysis rules and output constraints are enforced automatically. AWS Clean Rooms only returns the results that comply with the defined analysis rules.

AWS Clean Rooms supports SQL queries that can be different than other query engines. For specifications, see the AWS Clean Rooms SQL Reference. If you want to run queries on data tables protected with differential privacy, you should ensure that your queries are compatible with the general-purpose query structure of AWS Clean Rooms Differential Privacy.

> **Note**
>
> When using Cryptographic Computing for Clean Rooms, not all SQL operations generate valid results. For example, you can conduct a COUNT on an encrypted column but conducting a SUM on encrypted numbers leads to errors. In addition, queries might also yield incorrect results. For example, queries that SUM sealed columns produce errors. However, a GROUP BY query over sealed columns seems to succeed but produces different groups than those produced by a GROUP BY query over the cleartext.

The member paying for query compute costs is charged for the queries run in the collaboration.

The member who can query can select multiple members who can receive results to receive the results from a single query. For more information, see Querying configured tables using the SQL code editor. For general information about receiving query results, see Receiving and using analysis results.

Prerequisites

Before you run a SQL query, make sure that you have the following:

- An active membership in AWS Clean Rooms collaboration
- Access to at least one configured table in the collaboration
- Confirmation that the member responsible for query compute costs is an active collaboration member

For information about how to query data or view queries by calling the AWS Clean Rooms StartProtectedQuery API operation directly or by using the AWS SDKs, see the AWS Clean Rooms API Reference.

For information about query logging, see Analysis logging in AWS Clean Rooms.

> **Note**
>
> If you run a query on encrypted data tables, the results from the encrypted columns are encrypted.

Spark properties configuration for SQL queries

AWS Clean Rooms enables you to optionally customize Spark runtime behavior by configuring supported Spark properties for SQL queries. These properties let you fine-tune performance, memory usage, and query execution parameters. With this feature, you have greater control over how your Spark-based queries are processed, allowing for optimization based on your specific workload requirements.

You can now adjust settings such as shuffle partitions, broadcast join thresholds, and adaptive query execution parameters directly from the AWS Clean Rooms console. This feature is particularly useful for complex queries or large datasets where default configurations may not be optimal. By fine-tuning these Spark properties, you can potentially improve query performance, reduce resource consumption, and better manage memory usage for your Spark-based collaboration analyses.

To leverage this feature, you'll find a new **Spark properties** section in the query interface. You can select from a list of supported properties and specify custom values. You can also configure Spark properties programmatically using the StartProtectedQuery API. This advanced configuration option empowers data analysts and engineers to optimize their queries for enhanced efficiency and scalability.

For more information about Spark properties, including default values, see Spark Properties in the Apache Spark documentation.

The following topics explain how to query data in a collaboration using the AWS Clean Rooms console.

Topics

- Querying configured tables using the SQL code editor
- Querying ID mapping tables using the SQL code editor
- Querying configured tables using a SQL analysis template
- Querying with the analysis builder
- Viewing the impact of differential privacy
- Viewing recent queries
- Viewing query details
- Exporting query analysis logs

Querying configured tables using the SQL code editor

As a member who can query, you can build a query manually by writing SQL code in the SQL code editor. The SQL code editor is located in the **Analysis** section of the **Analysis** tab in the AWS Clean Rooms console.

The SQL code editor is displayed by default. If you want to use the analysis builder to build queries, see Querying with the analysis builder.

> **Important**
>
> If you start writing a SQL query in the code editor and then turn on the **Analysis builder UI**, your query isn't saved.

AWS Clean Rooms supports many SQL commands, functions, and conditions. For more information, see the AWS Clean Rooms SQL Reference.

> **Tip**
>
> If a scheduled maintenance occurs while a query is running, the query is terminated and rolled back. You must restart the query.

To query configured tables using the SQL code editor

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Run queries**.
4. On the **Analysis** tab, under **Tables**, view the list of tables and their associated analysis rule type (**Aggregation analysis rule**, **List analysis rule**, or **Custom analysis rule**).

> **Note**
>
> If you don't see the tables that you expect in the list, it might be for the following reasons:
>
> - The tables haven't been associated.
> - The tables don't have an analysis rule configured.

5. (Optional) To view the table's schema and analysis rule controls, expand the table by selecting the plus sign icon (**+**).
6. Under the **Analysis** section, for **Analysis mode**, select **Write SQL code**.

> **Note**
>
> The **Analysis** section only displays if the member who can receive results and the member who is responsible to pay for query compute costs have joined the collaboration as an active member.

7. Build the query by typing the query into the SQL code editor.

   For more information about supported SQL commands and functions, see the AWS Clean Rooms SQL Reference.

   You can also use the following options to build your query.

   Use an example query

   To use an example query

   1. Select the three vertical dots next to the table.
   2. Under **Insert in editor**, choose **Example query**.

> **Note**
>
> Inserting an **Example query** appends it to the query already in the editor.

The query example appears. All of the tables listed under **Tables** are included in the query.

3. Edit the placeholder values in the query.

Insert column names or functions

To insert a column name or function

1. Select the three vertical dots next to a column.
2. Under **Insert in editor**, choose **Column name**.
3. To manually insert a function that is permitted on a column,
   a. Select the three vertical dots next to a column.
   b. Select **Insert in editor**.
   c. Select the name of the permitted function (such as INNER JOIN, SUM, SUM DISTINCT, or COUNT).
4. Press **Ctrl** + **Space** to view the table schemas in the code editor.

> **Note**
>
> Members who can query can view and use the partition columns in each configured table association. Ensure the partition column is labeled as a partition column in the AWS Glue table underlying the configured table.

5. Edit the placeholder values in the query.

8. Specify the supported **Worker type** and the **Number of workers**.

   You can choose the instance type and number of instances (workers) to run your SQL queries.

   You can select 2 to 128 workers for CR.1X, 2 to 32 workers for CR.4X, and 2 to 16 workers for CR.8X.

Use the following table to determine the worker type and number of workers you need for your use case.

Worker type	vCPU	Memory (GB)	Storage (GB)	Number of workers	Total Clean Rooms Processing Units (CRPU)
CR.1X (default)	4	30	500	2	4
				128	256
CR.4X	16	120	2000	2	16
				32	256
CR.8X	32	244	2000	2	32
				16	256

> **Note**
>
> Different worker types and number of workers have associated costs. To learn more about the pricing, see AWS Clean Rooms pricing.

9. For **Send results to**, specify who can receive results.

> **Note**
>
> To receive results, the collaboration member must be configured as a result receiver and must be an active participant in the collaboration (**Status: Active**)

10. (Member who can query only) The **Use your default result settings** checkbox is selected by default. Keep this selected if you want to keep your default result settings.

If you want to specify different results settings for this query, clear the **Use your default result settings** checkbox, and then choose the following.

a. **Result format** (**CSV** or **PARQUET**)

b. **Result files** (**Single** or **Multiple**)

c. **Results destination in Amazon S3**

Each member who can receive results can specify a different **Result format**, **Result files**, and **Results destination in Amazon S3**.

11. To specify **Spark properties**:

    a. Expand **Spark properties**.

    b. Choose **Add Spark properties**.

    c. On the **Spark properties** dialog box, choose a **Property name** from the dropdown list and enter a **Value**.

The following tables provide a definition for each property.

For more information about Spark properties, see Spark Properties in the Apache Spark documentation.

> **Note**
>
> You can configure a maximum of 50 Spark properties. Each property value can be up to 500 characters.

Property Name	Description	Default Value
spark.task.maxFailures	Controls how many consecutive times a task can fail before the job fails. Requires a value greater than or equal to 1. The number of allowed retries equals this value minus 1. The failure count resets if any attempt	4

Property Name	Description	Default Value
	succeeds. Failures across different tasks don't accumulate toward this limit.	
spark.sql.files.maxPartitionBytes	Sets the maximum number of bytes to pack into a single partition when reading from file-based sources such as Parquet, JSON, and ORC.	128MB
spark.hadoop.fs.s3.maxRetries	Sets the maximum number of retry attempts for Amazon S3 file operations.	(none)
spark.network.timeout	Sets the default timeout for all network interactions. Overrides the following timeout settings if they aren't configured: • spark.storage.blockManagerHeartbeatTimeoutMs • spark.shuffle.io.connectionTimeout • spark.rpc.askTimeout • spark.rpc.lookupTimeout	120s
spark.rdd.compress	Specifies whether to compress serialized RDD partitions using spark.io.compression.codec. Applies to StorageLevel.MEMORY_ONLY_SER in Java and Scala, or StorageLevel.MEMORY_ONLY in Python. Reduces storage space but requires additional CPU processing time.	false
spark.shuffle.spill.compress	Specifies whether to compress shuffle spill data using spark.io.compression.codec.	true
spark.shuffle.compress	Specifies whether to compress map output files. Compression uses spark.io.compression.codec.	true

Property Name	Description	Default Value
spark.shuffle.service.index.cache.size	Sets the cache size limit, in bytes unless otherwise specified.	100m
spark.shuffle.io.maxRetries	Sets the maximum number of retries for fetches that fail due to IO-related exceptions.	3
spark.shuffle.io.retryWait	Sets the wait time between retries of fetches. The maximum delay caused by retrying is 15 seconds by default, calculated as maxRetries * retryWait.	5s
spark.shuffle.io.connectionTimeout	Sets the timeout for established connections between shuffle servers and clients to be marked as idle and closed if there are still outstanding fetch requests but no traffic on the channel.	(value of spark.network.timeout)
spark.driver.maxResultSize	Sets the total size limit of serialized results of all partitions for each Spark action, in bytes. Should be at least 1M, or 0 for unlimited.	1g
spark.memory.fraction	Sets the fraction of (heap space - 300MB) used for execution and storage. The lower this value, the more frequently spills and cached data eviction occur. Leaving this at the default value is recommended.	0.6
spark.scheduler.mode	Sets the scheduling mode between jobs submitted to the same SparkContext. Can be set to FAIR to use fair sharing instead of queueing jobs one after another. Supported values: FAIR, FIFO.	FIFO

Property Name	Description	Default Value
spark.sql.adaptive.advisoryPartitionSizeInBytes	Sets the target size in bytes for shuffle partitions during adaptive optimization when spark.sql.adaptive.enabled is true. Controls partition size when coalescing small partitions or splitting skewed partitions.	(value of spark.sql.adaptive.shuffle.targetPostShuffleInputSize)
spark.sql.adaptive.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Applies only in adaptive framework. Uses the same default value as spark.sql.autoBroadcastJoinThreshold. Set to -1 to disable broadcasting.	(none)
spark.sql.adaptive.coalescePartitions.enabled	Specifies whether to coalesce contiguous shuffle partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes to optimize task size. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Defines the initial number of shuffle partitions before coalescing. Requires both spark.sql.adaptive.enabled and spark.sql.adaptive.coalescePartitions.enabled to be true. Defaults to the value of spark.sql.shuffle.partitions.	(none)
spark.sql.adaptive.coalescePartitions.minPartitionSize	Sets the minimum size for coalesced shuffle partitions to prevent partitions from becoming too small during adaptive optimization.	1 MB

Property Name	Description	Default Value
spark.sql.adaptive.coalescePartitions.parallelismFirst	Specifies whether to calculate partition sizes based on cluster parallelism instead of spark.sql.adaptive.advisoryPartitionSizeInBytes during partition coalescing. Generates smaller partition sizes than the configured target size to maximize parallelism. We recommend setting this to false on busy clusters to improve resource utilization by preventing excessive small tasks.	true
spark.sql.adaptive.enabled	Specifies whether to enable adaptive query execution to re-optimize query plans during query execution, based on accurate runtime statistics.	true
spark.sql.adaptive.forceOptimizeSkewedJoin	Specifies whether to force enable OptimizeSkewedJoin even if it introduces extra shuffle.	false
spark.sql.adaptive.localShuffleReader.enabled	Specifies whether to use local shuffle readers when shuffle partitioning isn't required, such as after converting from sort-merge joins to broadcast-hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Sets the maximum partition size in bytes for building local hash maps. Prioritizes shuffled hash joins over sort-merge joins when: • This value equals or exceeds spark.sql.adaptive.advisoryPartitionSizeInBytes • All partition sizes are within this limit Overrides spark.sql.join.preferSortMergeJoin setting.	0 bytes
spark.sql.adaptive.optimizeSkewsInRebalancePartitions.enabled	Specifies whether to optimize skewed shuffle partitions by splitting them into smaller partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.rebalancePartitionsSmallPartitionFactor	Defines the size threshold factor for merging partitions during splitting. Partitions smaller than this factor multiplied by spark.sql.adaptive.advisoryPartitionSizeInBytes are merged.	0.2
spark.sql.adaptive.skewJoin.enabled	Specifies whether to handle data skew in shuffled joins by splitting and optionally replicating skewed partitions. Applies to sort-merge and shuffled hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Determines the size factor that determines partition skew. A partition is skewed when its size exceeds both: • This factor multiplied by the median partition size • The value of spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Sets the size threshold in bytes for identifying skewed partitions. A partition is skewed when its size exceeds both: • This threshold • The median partition size multiplied by spark.sql.adaptive.skewJoin.skewedPartitionFactor We recommend setting this value larger than spark.sql.adaptive.advisoryPartitionSizeInBytes.	256MB
spark.sql.broadcastTimeout	Controls the timeout period in seconds for the broadcast operations during broadcast joins.	300 seconds
spark.sql.cbo.enabled	Specifies whether to enable cost-based optimization (CBO) for plan statistics estimation.	false
spark.sql.cbo.joinReorder.dp.star.filter	Specifies whether to apply star-join filter heuristics during cost-based join enumeration.	false

Property Name	Description	Default Value
spark.sql.cbo.joinReorder.dp.threshold	Sets the maximum number of joined nodes allowed in the dynamic programming algorithm.	12
spark.sql.cbo.joinReorder.enabled	Specifies whether to enable join reordering in cost-based optimization (CBO).	false
spark.sql.cbo.planStats.enabled	Specifies whether to fetch row counts and column statistics from the catalog during logical plan generation.	false
spark.sql.cbo.starSchemaDetection	Specifies whether to enable join reordering based on star schema detection.	false
spark.sql.files.maxPartitionNum	Sets the target maximum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Rescales partitions when the initial count exceeds this value. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.files.maxRecordsPerFile	Sets the maximum number of records to write to a single file. No limit applies when set to zero or a negative value.	0
spark.sql.files.minPartitionNum	Sets the target minimum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Defaults to spark.sql.leafNodeDefaultParallelism. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.inMemoryColumnarStorage.batchSize	Controls the batch size for columnar caching. Increasing the size improves memory utilization and compression but increases the risk of out-of-memory errors.	10000

Property Name	Description	Default Value
spark.sql.inMemoryColumnarStorage.compressed	Specifies whether to automatically select compression codecs for columns based on data statistics.	true
spark.sql.inMemoryColumnarStorage.enableVectorizedReader	Specifies whether to enable vectorized reading for columnar caching.	true
spark.sql.legacy.allowHashOnMapType	Specifies whether to allow hash operations on map type data structures. This legacy setting maintains compatibility with older Spark versions' map type handling.	(none)
spark.sql.legacy.allowNegativeScaleOfDecimal	Specifies whether to allow negative scale values in decimal type definitions. This legacy setting maintains compatibility with older Spark versions that supported negative decimal scales.	(none)
spark.sql.legacy.castComplexTypesToString.enabled	Specifies whether to enable legacy behavior for casting complex types to strings. Maintains compatibility with older Spark versions' type conversion rules.	(none)
spark.sql.legacy.charVarcharAsString	Specifies whether to treat CHAR and VARCHAR types as STRING types. This legacy setting provides compatibility with older Spark versions' string type handling.	(none)
spark.sql.legacy.createEmptyCollectionUsingStringType	Specifies whether to create empty collections using string type elements. This legacy setting maintains compatibility with older Spark versions' collection initialization behavior.	(none)

Property Name	Description	Default Value
spark.sql.legacy.exponentLiteralAsDecimal.enabled	Specifies whether to interpret exponential literals as decimal types. This legacy setting maintains compatibility with older Spark versions' numeric literal handling.	(none)
spark.sql.legacy.json.allowEmptyString.enabled	Specifies whether to allow empty strings in JSON processing. This legacy setting maintains compatibility with older Spark versions' JSON parsing behavior.	(none)
spark.sql.legacy.parquet.int96RebaseModeInRead	Specifies whether to use legacy INT96 timestamp rebase mode when reading Parquet files. This legacy setting maintains compatibility with older Spark versions' timestamp handling.	(none)
spark.sql.legacy.timeParserPolicy	Controls the time parsing behavior for backwards compatibility. This legacy setting determines how timestamps and dates are parsed from strings.	(none)
spark.sql.legacy.typeCoercion.datetimeToString.enabled	Specifies whether to enable legacy type coercion behavior when converting datetime values to strings. Maintains compatibility with older Spark versions' datetime conversion rules.	(none)
spark.sql.maxSinglePartitionBytes	Sets the maximum partition size in bytes. The planner introduces shuffle operations for larger partitions to improve parallelism.	128m

Property Name	Description	Default Value
spark.sql.metadataCacheTTLSeconds	Controls the time-to-live (TTL) for metadata caches. Applies to partition file metadata and session catalog caches. Requires: • A positive value greater than zero • spark.sql.catalogImplementation set to hive • spark.sql.hive.filesourcePartitionFileCacheSize greater than zero • spark.sql.hive.manageFilesourcePartitions set to true	-1000ms
spark.sql.optimizer.collapseProjectAlwaysInline	Specifies whether to collapse adjacent projections and inline expressions, even when it causes duplication.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	Specifies whether to generate predicates for partition columns used as join keys.	true
spark.sql.optimizer.enableCsvExpressionOptimization	Specifies whether to optimize CSV expressions in SQL optimizer by pruning unnecessary columns from from_csv operations.	true
spark.sql.optimizer.enableJsonExpressionOptimization	Specifies whether to optimize JSON expressions in SQL optimizer by: • Pruning unnecessary columns from from_json operations • Simplifying from_json and to_json combinations • Optimizing named_struct operations	true

Property Name	Description	Default Value
spark.sql.optimizer.excludedRules	Defines optimizer rules to disable, identified by comma-separated rule names. Some rules cannot be disabled as they are required for correctness. The optimizer logs which rules are successfully disabled.	(none)
spark.sql.optimizer.runtime.bloomFilter.applicationSideScanSizeThreshold	Sets the minimum aggregated scan size in bytes required to inject a Bloom filter on the application side.	10GB
spark.sql.optimizer.runtime.bloomFilter.creationSideThreshold	Defines the maximum size threshold for injecting a Bloom filter on the creation side.	10MB
spark.sql.optimizer.runtime.bloomFilter.enabled	Specifies whether to insert a Bloom filter to reduce shuffle data when one side of a shuffle join has a selective predicate.	true
spark.sql.optimizer.runtime.bloomFilter.expectedNumItems	Defines the default number of expected items in the runtime Bloom filter.	1000000
spark.sql.optimizer.runtime.bloomFilter.maxNumBits	Sets the maximum number of bits allowed in the runtime Bloom filter.	67108864
spark.sql.optimizer.runtime.bloomFilter.maxNumItems	Sets the maximum number of expected items allowed in the runtime Bloom filter.	4000000
spark.sql.optimizer.runtime.bloomFilter.numBits	Defines the default number of bits used in the runtime Bloom filter.	8388608

Property Name	Description	Default Value
spark.sql.optimizer.runtime.rowLevelOperationGroupFilter.enabled	Specifies whether to enable runtime group filtering for row-level operations. Allows data sources to: • Prune entire groups of data (such as files or partitions) using data source filters • Execute runtime queries to identify matching records • Discard unnecessary groups to avoid expensive rewrites Limitations: • Not all expressions can convert to data source filters • Some expressions require Spark evaluation (such as subqueries)	true
spark.sql.optimizer.runtimeFilter.number.threshold	Sets the total number of injected runtime filters (non-DPP). This is to prevent driver OOMs with too many Bloom filters.	10
spark.sql.optimizer.runtimeFilter.semiJoinReduction.enabled	Specifies whether to insert a semi-join to reduce shuffle data when one side of a shuffle join has a selective predicate.	false

Property Name	Description	Default Value
spark.sql.parquet.aggregatePushdown	Specifies whether to push down aggregates to Parquet for optimization. Supports: • MIN and MAX for boolean, integer, float, and date types • COUNT for all data types Throws an exception if statistics are missing from any Parquet file footer.	false
spark.sql.parquet.columnarReaderBatchSize	Controls the number of rows in each Parquet vectorized reader batch. Choose a value that balances performance overhead and memory usage to prevent out-of-memory errors.	4096
spark.sql.parquet.enableVectorizedReader	Specifies whether to enable vectorized Parquet decoding.	true
spark.sql.shuffle.partitions	Sets the default number of partitions for data shuffling during joins or aggregations. Cannot be modified between structured streaming query restarts from the same checkpoint location.	200
spark.sql.shuffledHashJoinFactor	Defines the multiplication factor used to determine shuffle hash join eligibility. A shuffle hash join is selected when the small-side data size multiplied by this factor is less than the large-side data size.	3

Property Name	Description	Default Value
spark.sql.sources.parallelPartitionDiscovery.threshold	Sets the maximum number of paths for driver-side file listing with file-based sources (Parquet, JSON, and ORC). When exceeded during partition discovery, files are listed using a separate Spark distributed job.	32
spark.sql.statistics.histogram.enabled	Specifies whether to generate equi-height histograms during column statistics computation to improve estimation accuracy. Requires an additional table scan beyond the one needed for basic column statistics.	false
spark.dynamicAllocation.executorIdleTimeout	Sets the duration an executor must be idle before it is removed when dynamic allocation is enabled.	60s
spark.dynamicAllocation.schedulerBacklogTimeout	Sets the duration that pending tasks must be backlogged before new executors are requested when dynamic allocation is enabled.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Same as spark.dynamicAllocation.schedulerBacklogTimeout, but used only for subsequent executor requests.	(value of spark.dynamicAllocation.schedulerBacklogTimeout)

Property Name	Description	Default Value
spark.scheduler.minRegisteredResourcesRatio	Sets the minimum ratio of registered resources (registered resources / total expected resources) to wait for before scheduling begins. Specified as a double between 0.0 and 1.0. Regardless of whether the minimum ratio of resources has been reached, the maximum amount of time it will wait before scheduling begins is controlled by spark.scheduler.maxRegisteredResourcesWaitingTime.	0.8
spark.scheduler.maxRegisteredResourcesWaitingTime	Sets the maximum amount of time to wait for resources to register before scheduling begins.	30s
spark.sql.hive.metastorePartitionPruningFallbackOnException	Specifies whether to fall back to getting all partitions from Hive metastore and perform partition pruning on the Spark client side when encountering MetaException from the metastore.	false
spark.sql.crossJoin.enabled	Specifies whether to allow queries that contain a cartesian product without explicit CROSS JOIN syntax.	true
spark.sql.analyzer.maxIterations	Sets the maximum number of iterations the query analyzer runs before giving up. Higher values allow the analyzer to process very large or deeply nested queries.	100
spark.sql.datapreffetch.filescan.maxParallelismPerTask	Sets the maximum number of file splits to pre-fetch concurrently for each task when scanning files.	4

Property Name	Description	Default Value
spark.sql.iceberg.data-prefetch.enabled	Specifies whether to enable data pre-fetch optimization when reading Iceberg tables.	true
spark.sql.legacy.nullValueWrittenAsQuotedEmptyStringCsv	Specifies whether to restore the legacy behavior of writing nulls as quoted empty strings in CSV output. When false, Spark writes nulls as unquoted empty strings.	false
spark.maxRemoteBlockSizeFetchToMem	Sets the size threshold above which Spark fetches remote blocks to disk instead of memory. This avoids a single large request consuming too much memory.	200m
spark.emr-serverless.allocation.batch.size	Sets the number of executors to request at once in each round of executor allocation.	20

Property Name	Description	Default Value
spark.sql.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Set to -1 to disable broadcasting.	10MB (-1 for CR.4X with 32 workers)
spark.dynamicAllocation.enabled	Specifies whether to use dynamic resource allocation, which scales the number of executors registered with this application up and down based on the workload.	true

Property Name	Description	Default Value
spark.files.fetchFailure.unRegisterOutputOnHost	Specifies whether to unregister all map outputs on a host when a fetch failure occurs. When false, Spark unregisters only the outputs of the specific executor that failed, which reduces unnecessary stage recomputation.	false
spark.io.compression.codec	Sets the codec used to compress internal data such as RDD partitions, event log, broadcast variables, and shuffle outputs. Supported values: lz4, snappy, zstd, gzip.	snappy
spark.sql.session.timeZone	Defines the session time zone for handling timestamps in string literals and Java object conversion. Accepts: • Region-based IDs in area/city format (such as America/Los_Angeles) • Zone offsets in (+/-)HH, (+/-)HH:mm, or (+/-)HH:mm:ss format (such as -08 or +01:00) • UTC or Z as aliases for +00:00	UTC

12. (Optional) For **Compute payer**, select the collaboration member who pays for query compute costs.

> **Note**
>
> If there is only one payer candidate for query compute in the collaboration, it defaults to that payer.

13. Choose **Run**.

> **Note**
>
> You can't run the query if the member who can receive results hasn't configured the query results settings.

14. View the **Results**.

    For more information, see Receiving and using analysis results.

15. Continue to adjust parameters and run your query again, or choose the **+** button to start a new query in a new tab.

> **Note**
>
> AWS Clean Rooms aims to provide clear error messaging. If an error message doesn't have enough details to help you troubleshoot, contact the account team. Provide them with a description of how the error occurred and the error message (including any identifiers). For more information, see Troubleshooting AWS Clean Rooms.

Querying ID mapping tables using the SQL code editor

The following procedure describes how to run a multi-table join query on the ID mapping table to join the `sourceId` with the `targetId`.

Before you query the ID mapping table, the ID mapping table must be successfully populated.

To query ID mapping tables using the SQL code editor

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Run queries**.
4. On the **Analysis** tab, go to the **Analysis** section.

> **Note**
>
> The **Analysis** section only displays if the member who can receive results and the member who is responsible to pay for query compute costs have joined the collaboration as an active member.

5. On the **Analysis** tab, under **Tables**, view the list of ID mapping tables (under **Managed by AWS Clean Rooms**) and their associated analysis rule type (**ID mapping table analysis rule**).

> **Note**
>
> If you don't see the ID mapping tables that you expect in the list, it might be because the ID mapping tables haven't been successfully populated. For more information, see Populating an existing ID mapping table.

6. Build the query by typing the query into the SQL code editor.

(Optional) If you want to use an example query	(Optional) If you want to insert a table name
1. Select the three vertical dots next to the table. 2. Under **Insert in editor**, choose **Example JOIN statement**. **Note** Inserting an **Example JOIN statements** appends the query already in the editor. The example JOIN statement appears. 3. Edit the placeholder values in the query.	1. Select the three vertical dots next to a column. 2. Under **Insert in editor**, choose **Table name**. 3. Edit the placeholder values in the query.

7. Specify the supported **Worker type** and the **Number of workers**.

Use the following table to determine the worker type and number of workers you need for your use case.

Worker type	vCPU	Memory (GB)	Storage (GB)	Number of workers	Total Clean Rooms Processing Units (CRPU)
CR.1X (default)	4	30	500	2	4
				128	256
CR.4X	16	120	2000	2	16
				32	256
CR.8X	32	244	2000	2	32
				16	256

> **Note**
>
> Different worker types and number of workers have associated costs. To learn more about the pricing, see AWS Clean Rooms pricing.

8. Choose **Run**.

> **Note**
>
> You can't run the query if the member who can receive results hasn't configured the query results settings.

9. View the **Results**.

   For more information, see Receiving and using analysis results.

10. Continue to adjust parameters and run your query again, or choose the **+** button to start a new query in a new tab.

> **Note**
>
> AWS Clean Rooms aims to provide clear error messaging. If an error message doesn't have enough details to help you troubleshoot, contact the account team. Provide them with a description of how the error occurred and the error message (including any identifiers). For more information, see Troubleshooting AWS Clean Rooms.

Querying configured tables using a SQL analysis template

This procedure demonstrates how to use an analysis template in the AWS Clean Rooms console to query configured tables with the **Custom** analysis rule.

To use a SQL analysis template to query configured tables with the Custom analysis rule

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Run queries**.
4. On the **Analysis** tab, under the **Tables** section, view the tables and their associated analysis rule type (**Custom analysis rule**).

   > **Note**
   >
   > If you don’t see the tables that you expect in the list, it might be for the following reasons:
   >
   > - The tables haven't been associated.
   > - The tables don't have an analysis rule configured.

5. Under the **Analysis** section, for **Analysis mode**, select **Run analysis templates** and then choose the analysis template from the dropdown list.
6. The parameters form the SQL analysis template will automatically populate in the **Definition**.
7. Specify the supported **Worker type** and the **Number of workers**.

Use the following table to determine the worker type and number of workers you need for your use case.

Worker type	vCPU	Memory (GB)	Storage (GB)	Number of workers	Total Clean Rooms Processing Units (CRPU)
CR.1X (default)	4	30	500	2	4
				128	256
CR.4X	16	120	2000	2	16
				32	256
CR.8X	32	244	2000	2	32
				16	256

> **Note**
>
> Different worker types and number of workers have associated costs. To learn more about the pricing, see AWS Clean Rooms pricing.

8. Specify the supported **Spark properties**.

   a. Select **Add Spark properties**.

   b. On the **Spark properties** dialog box, choose a **Property name** from the dropdown list and enter a **Value**.

The following tables provide a definition for each property.

For more information about Spark properties, see Spark Properties in the Apache Spark documentation.

> **Note**
>
> You can configure a maximum of 50 Spark properties. Each property value can be up to 500 characters.

Property Name	Description	Default Value
spark.task.maxFailures	Controls how many consecutive times a task can fail before the job fails. Requires a value greater than or equal to 1. The number of allowed retries equals this value minus 1. The failure count resets if any attempt succeeds. Failures across different tasks don't accumulate toward this limit.	4
spark.sql.files.maxPartitionBytes	Sets the maximum number of bytes to pack into a single partition when reading from file-based sources such as Parquet, JSON, and ORC.	128MB
spark.hadoop.fs.s3.maxRetries	Sets the maximum number of retry attempts for Amazon S3 file operations.	(none)
spark.network.timeout	Sets the default timeout for all network interactions. Overrides the following timeout settings if they aren't configured: • spark.storage.blockManagerHeartbeatTimeoutMs • spark.shuffle.io.connectionTimeout • spark.rpc.askTimeout • spark.rpc.lookupTimeout	120s

Property Name	Description	Default Value
spark.rdd.compress	Specifies whether to compress serialized RDD partitions using spark.io.compression.codec. Applies to StorageLevel.MEMORY_ONLY_SER in Java and Scala, or StorageLevel.MEMORY_ONLY in Python. Reduces storage space but requires additional CPU processing time.	false
spark.shuffle.spill.compress	Specifies whether to compress shuffle spill data using spark.io.compression.codec.	true
spark.shuffle.compress	Specifies whether to compress map output files. Compression uses spark.io.compression.codec.	true
spark.shuffle.service.index.cache.size	Sets the cache size limit, in bytes unless otherwise specified.	100m
spark.shuffle.io.maxRetries	Sets the maximum number of retries for fetches that fail due to IO-related exceptions.	3
spark.shuffle.io.retryWait	Sets the wait time between retries of fetches. The maximum delay caused by retrying is 15 seconds by default, calculated as maxRetries * retryWait.	5s
spark.shuffle.io.connectionTimeout	Sets the timeout for established connections between shuffle servers and clients to be marked as idle and closed if there are still outstanding fetch requests but no traffic on the channel.	(value of spark.network.timeout)

Property Name	Description	Default Value
spark.driver.maxResultSize	Sets the total size limit of serialized results of all partitions for each Spark action, in bytes. Should be at least 1M, or 0 for unlimited.	1g
spark.memory.fraction	Sets the fraction of (heap space - 300MB) used for execution and storage. The lower this value, the more frequently spills and cached data eviction occur. Leaving this at the default value is recommended.	0.6
spark.scheduler.mode	Sets the scheduling mode between jobs submitted to the same SparkContext. Can be set to FAIR to use fair sharing instead of queueing jobs one after another. Supported values: FAIR, FIFO.	FIFO
spark.sql.adaptive.advisoryPartitionSizeInBytes	Sets the target size in bytes for shuffle partitions during adaptive optimization when spark.sql.adaptive.enabled is true. Controls partition size when coalescing small partitions or splitting skewed partitions.	(value of spark.sql.adaptive.shuffle.targetPostShuffleInputSize)
spark.sql.adaptive.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Applies only in adaptive framework. Uses the same default value as spark.sql.autoBroadcastJoinThreshold. Set to -1 to disable broadcasting.	(none)
spark.sql.adaptive.coalescePartitions.enabled	Specifies whether to coalesce contiguous shuffle partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes to optimize task size. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Defines the initial number of shuffle partitions before coalescing. Requires both spark.sql.adaptive.enabled and spark.sql.adaptive.coalescePartitions.enabled to be true. Defaults to the value of spark.sql.shuffle.partitions.	(none)
spark.sql.adaptive.coalescePartitions.minPartitionSize	Sets the minimum size for coalesced shuffle partitions to prevent partitions from becoming too small during adaptive optimization.	1 MB
spark.sql.adaptive.coalescePartitions.parallelismFirst	Specifies whether to calculate partition sizes based on cluster parallelism instead of spark.sql.adaptive.advisoryPartitionSizeInBytes during partition coalescing. Generates smaller partition sizes than the configured target size to maximize parallelism. We recommend setting this to false on busy clusters to improve resource utilization by preventing excessive small tasks.	true
spark.sql.adaptive.enabled	Specifies whether to enable adaptive query execution to re-optimize query plans during query execution, based on accurate runtime statistics.	true
spark.sql.adaptive.forceOptimizeSkewedJoin	Specifies whether to force enable OptimizeSkewedJoin even if it introduces extra shuffle.	false

Property Name	Description	Default Value
spark.sql.adaptive.localShuffleReader.enabled	Specifies whether to use local shuffle readers when shuffle partitioning isn't required, such as after converting from sort-merge joins to broadcast-hash joins. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Sets the maximum partition size in bytes for building local hash maps. Prioritizes shuffled hash joins over sort-merge joins when: • This value equals or exceeds spark.sql.adaptive.advisoryPartitionSizeInBytes • All partition sizes are within this limit Overrides spark.sql.join.preferSortMergeJoin setting.	0 bytes
spark.sql.adaptive.optimizeSkewsInRebalancePartitions.enabled	Specifies whether to optimize skewed shuffle partitions by splitting them into smaller partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.rebalancePartitionsSmallPartitionFactor	Defines the size threshold factor for merging partitions during splitting. Partitions smaller than this factor multiplied by spark.sql.adaptive.advisoryPartitionSizeInBytes are merged.	0.2

Property Name	Description	Default Value
spark.sql.adaptive.skewJoin.enabled	Specifies whether to handle data skew in shuffled joins by splitting and optionally replicating skewed partitions. Applies to sort-merge and shuffled hash joins. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Determines the size factor that determines partition skew. A partition is skewed when its size exceeds both: • This factor multiplied by the median partition size • The value of spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Sets the size threshold in bytes for identifying skewed partitions. A partition is skewed when its size exceeds both: • This threshold • The median partition size multiplied by spark.sql.adaptive.skewJoin.skewedPartitionFactor We recommend setting this value larger than spark.sql.adaptive.advisoryPartitionSizeInBytes.	256MB
spark.sql.broadcastTimeout	Controls the timeout period in seconds for the broadcast operations during broadcast joins.	300 seconds
spark.sql.cbo.enabled	Specifies whether to enable cost-based optimization (CBO) for plan statistics estimation.	false

Property Name	Description	Default Value
spark.sql.cbo.joinReorder.dp.star.filter	Specifies whether to apply star-join filter heuristics during cost-based join enumeration.	false
spark.sql.cbo.joinReorder.dp.threshold	Sets the maximum number of joined nodes allowed in the dynamic programming algorithm.	12
spark.sql.cbo.joinReorder.enabled	Specifies whether to enable join reordering in cost-based optimization (CBO).	false
spark.sql.cbo.planStats.enabled	Specifies whether to fetch row counts and column statistics from the catalog during logical plan generation.	false
spark.sql.cbo.starSchemaDetection	Specifies whether to enable join reordering based on star schema detection.	false
spark.sql.files.maxPartitionNum	Sets the target maximum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Rescales partitions when the initial count exceeds this value. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.files.maxRecordsPerFile	Sets the maximum number of records to write to a single file. No limit applies when set to zero or a negative value.	0
spark.sql.files.minPartitionNum	Sets the target minimum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Defaults to spark.sql.leafNodeDefaultParallelism. This is a suggested target, not a guaranteed limit.	(none)

Property Name	Description	Default Value
spark.sql.inMemoryColumnarStorage.batchSize	Controls the batch size for columnar caching. Increasing the size improves memory utilization and compression but increases the risk of out-of-memory errors.	10000
spark.sql.inMemoryColumnarStorage.compressed	Specifies whether to automatically select compression codecs for columns based on data statistics.	true
spark.sql.inMemoryColumnarStorage.enableVectorizedReader	Specifies whether to enable vectorized reading for columnar caching.	true
spark.sql.legacy.allowHashOnMapType	Specifies whether to allow hash operations on map type data structures. This legacy setting maintains compatibility with older Spark versions' map type handling.	(none)
spark.sql.legacy.allowNegativeScaleOfDecimal	Specifies whether to allow negative scale values in decimal type definitions. This legacy setting maintains compatibility with older Spark versions that supported negative decimal scales.	(none)
spark.sql.legacy.castComplexTypesToString.enabled	Specifies whether to enable legacy behavior for casting complex types to strings. Maintains compatibility with older Spark versions' type conversion rules.	(none)
spark.sql.legacy.charVarcharAsString	Specifies whether to treat CHAR and VARCHAR types as STRING types. This legacy setting provides compatibility with older Spark versions' string type handling.	(none)

Property Name	Description	Default Value
spark.sql.legacy.createEmptyCollectionUsingStringType	Specifies whether to create empty collections using string type elements. This legacy setting maintains compatibility with older Spark versions' collection initialization behavior.	(none)
spark.sql.legacy.exponentLiteralAsDecimal.enabled	Specifies whether to interpret exponential literals as decimal types. This legacy setting maintains compatibility with older Spark versions' numeric literal handling.	(none)
spark.sql.legacy.json.allowEmptyString.enabled	Specifies whether to allow empty strings in JSON processing. This legacy setting maintains compatibility with older Spark versions' JSON parsing behavior.	(none)
spark.sql.legacy.parquet.int96RebaseModeInRead	Specifies whether to use legacy INT96 timestamp rebase mode when reading Parquet files. This legacy setting maintains compatibility with older Spark versions' timestamp handling.	(none)
spark.sql.legacy.timeParserPolicy	Controls the time parsing behavior for backwards compatibility. This legacy setting determines how timestamps and dates are parsed from strings.	(none)
spark.sql.legacy.typeCoercion.datetimeToString.enabled	Specifies whether to enable legacy type coercion behavior when converting datetime values to strings. Maintains compatibility with older Spark versions' datetime conversion rules.	(none)
spark.sql.maxSinglePartitionBytes	Sets the maximum partition size in bytes. The planner introduces shuffle operations for larger partitions to improve parallelism.	128m

Property Name	Description	Default Value
spark.sql.metadataCacheTTLSeconds	Controls the time-to-live (TTL) for metadata caches. Applies to partition file metadata and session catalog caches. Requires: • A positive value greater than zero • spark.sql.catalogImplementation set to hive • spark.sql.hive.filesourcePartitionFileCacheSize greater than zero • spark.sql.hive.manageFilesourcePartitions set to true	-1000ms
spark.sql.optimizer.collapseProjectAlwaysInline	Specifies whether to collapse adjacent projections and inline expressions, even when it causes duplication.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	Specifies whether to generate predicates for partition columns used as join keys.	true
spark.sql.optimizer.enableCsvExpressionOptimization	Specifies whether to optimize CSV expressions in SQL optimizer by pruning unnecessary columns from from_csv operations.	true
spark.sql.optimizer.enableJsonExpressionOptimization	Specifies whether to optimize JSON expressions in SQL optimizer by: • Pruning unnecessary columns from from_json operations • Simplifying from_json and to_json combinations • Optimizing named_struct operations	true

Property Name	Description	Default Value
spark.sql.optimizer.excludedRules	Defines optimizer rules to disable, identified by comma-separated rule names. Some rules cannot be disabled as they are required for correctness. The optimizer logs which rules are successfully disabled.	(none)
spark.sql.optimizer.runtime.bloomFilter.applicationSideScanSizeThreshold	Sets the minimum aggregated scan size in bytes required to inject a Bloom filter on the application side.	10GB
spark.sql.optimizer.runtime.bloomFilter.creationSideThreshold	Defines the maximum size threshold for injecting a Bloom filter on the creation side.	10MB
spark.sql.optimizer.runtime.bloomFilter.enabled	Specifies whether to insert a Bloom filter to reduce shuffle data when one side of a shuffle join has a selective predicate.	true
spark.sql.optimizer.runtime.bloomFilter.expectedNumItems	Defines the default number of expected items in the runtime Bloom filter.	1000000
spark.sql.optimizer.runtime.bloomFilter.maxNumBits	Sets the maximum number of bits allowed in the runtime Bloom filter.	67108864
spark.sql.optimizer.runtime.bloomFilter.maxNumItems	Sets the maximum number of expected items allowed in the runtime Bloom filter.	4000000
spark.sql.optimizer.runtime.bloomFilter.numBits	Defines the default number of bits used in the runtime Bloom filter.	8388608

Property Name	Description	Default Value
spark.sql.optimizer.runtime.rowLevelOperationGroupFilter.enabled	Specifies whether to enable runtime group filtering for row-level operations. Allows data sources to: • Prune entire groups of data (such as files or partitions) using data source filters • Execute runtime queries to identify matching records • Discard unnecessary groups to avoid expensive rewrites Limitations: • Not all expressions can convert to data source filters • Some expressions require Spark evaluation (such as subqueries)	true
spark.sql.optimizer.runtimeFilter.number.threshold	Sets the total number of injected runtime filters (non-DPP). This is to prevent driver OOMs with too many Bloom filters.	10
spark.sql.optimizer.runtimeFilter.semiJoinReduction.enabled	Specifies whether to insert a semi-join to reduce shuffle data when one side of a shuffle join has a selective predicate.	false

Property Name	Description	Default Value
spark.sql.parquet.aggregatePushdown	Specifies whether to push down aggregates to Parquet for optimization. Supports: • MIN and MAX for boolean, integer, float, and date types • COUNT for all data types Throws an exception if statistics are missing from any Parquet file footer.	false
spark.sql.parquet.columnarReaderBatchSize	Controls the number of rows in each Parquet vectorized reader batch. Choose a value that balances performance overhead and memory usage to prevent out-of-memory errors.	4096
spark.sql.parquet.enableVectorizedReader	Specifies whether to enable vectorized Parquet decoding.	true
spark.sql.shuffle.partitions	Sets the default number of partitions for data shuffling during joins or aggregations. Cannot be modified between structured streaming query restarts from the same checkpoint location.	200
spark.sql.shuffledHashJoinFactor	Defines the multiplication factor used to determine shuffle hash join eligibility. A shuffle hash join is selected when the small-side data size multiplied by this factor is less than the large-side data size.	3

Property Name	Description	Default Value
spark.sql.sources.parallelPartitionDiscovery.threshold	Sets the maximum number of paths for driver-side file listing with file-based sources (Parquet, JSON, and ORC). When exceeded during partition discovery, files are listed using a separate Spark distributed job.	32
spark.sql.statistics.histogram.enabled	Specifies whether to generate equi-height histograms during column statistics computation to improve estimation accuracy. Requires an additional table scan beyond the one needed for basic column statistics.	false
spark.dynamicAllocation.executorIdleTimeout	Sets the duration an executor must be idle before it is removed when dynamic allocation is enabled.	60s
spark.dynamicAllocation.schedulerBacklogTimeout	Sets the duration that pending tasks must be backlogged before new executors are requested when dynamic allocation is enabled.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Same as spark.dynamicAllocation.schedulerBacklogTimeout, but used only for subsequent executor requests.	(value of spark.dynamicAllocation.schedulerBacklogTimeout)

Property Name	Description	Default Value
spark.scheduler.minRegisteredResourcesRatio	Sets the minimum ratio of registered resources (registered resources / total expected resources) to wait for before scheduling begins. Specified as a double between 0.0 and 1.0. Regardless of whether the minimum ratio of resources has been reached, the maximum amount of time it will wait before scheduling begins is controlled by spark.scheduler.maxRegisteredResourcesWaitingTime.	0.8
spark.scheduler.maxRegisteredResourcesWaitingTime	Sets the maximum amount of time to wait for resources to register before scheduling begins.	30s
spark.sql.hive.metastorePartitionPruningFallbackOnException	Specifies whether to fall back to getting all partitions from Hive metastore and perform partition pruning on the Spark client side when encountering MetaException from the metastore.	false
spark.sql.crossJoin.enabled	Specifies whether to allow queries that contain a cartesian product without explicit CROSS JOIN syntax.	true
spark.sql.analyzer.maxIterations	Sets the maximum number of iterations the query analyzer runs before giving up. Higher values allow the analyzer to process very large or deeply nested queries.	100
spark.sql.datapreffetch.filescan.maxParallelismPerTask	Sets the maximum number of file splits to pre-fetch concurrently for each task when scanning files.	4

Property Name	Description	Default Value
spark.sql.iceberg.data-prefetch.enabled	Specifies whether to enable data pre-fetch optimization when reading Iceberg tables.	true
spark.sql.legacy.nullValueWrittenAsQuotedEmptyStringCsv	Specifies whether to restore the legacy behavior of writing nulls as quoted empty strings in CSV output. When false, Spark writes nulls as unquoted empty strings.	false
spark.maxRemoteBlockSizeFetchToMem	Sets the size threshold above which Spark fetches remote blocks to disk instead of memory. This avoids a single large request consuming too much memory.	200m
spark.emr-serverless.allocation.batch.size	Sets the number of executors to request at once in each round of executor allocation.	20

Property Name	Description	Default Value
spark.sql.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Set to -1 to disable broadcasting.	10MB (-1 for CR.4X with 32 workers)
spark.dynamicAllocation.enabled	Specifies whether to use dynamic resource allocation, which scales the number of executors registered with this application up and down based on the workload.	true

Property Name	Description	Default Value
spark.files.fetchFailure.unRegisterOutputOnHost	Specifies whether to unregister all map outputs on a host when a fetch failure occurs. When false, Spark unregisters only the outputs of the specific executor that failed, which reduces unnecessary stage recomputation.	false
spark.io.compression.codec	Sets the codec used to compress internal data such as RDD partitions, event log, broadcast variables, and shuffle outputs. Supported values: lz4, snappy, zstd, gzip.	snappy
spark.sql.session.timeZone	Defines the session time zone for handling timestamps in string literals and Java object conversion. Accepts: • Region-based IDs in area/city format (such as America/Los_Angeles) • Zone offsets in (+/-)HH, (+/-)HH:mm, or (+/-)HH:mm:ss format (such as -08 or +01:00) • UTC or Z as aliases for +00:00	UTC

9. (Optional) For **Compute payer**, select the collaboration member who pays for query compute costs.

> **Note**
>
> If there is only one payer candidate for query compute in the collaboration, it defaults to that payer.

10. Choose **Run**.

> **Note**
>
> You can't run the query if the member who can receive results hasn't configured the query results settings.

11. Continue to adjust parameters and run your query again, or choose the **+** button to start a new query in a new tab.

Querying with the analysis builder

You can use the analysis builder to build queries without having to write SQL code. With the analysis builder, you can build a query for a collaboration that has:

- A single table that uses the aggregation analysis rule with no JOIN required
- Two tables (one from each member) that both use the aggregation analysis rule
- Two tables (one from each member) that both use the list analysis rule
- Two tables (one from each member) that both use the aggregation analysis rule and two tables (one from each member) that both use the list analysis rule

If you want to manually write SQL queries, see Querying configured tables using the SQL code editor.

The analysis builder appears as the **Analysis builder UI** option in the **Analysis** section of the **Analysis** tab in the AWS Clean Rooms console.

> **Important**
>
> If you turn on the **Analysis builder UI**, start building a query in the analysis builder, and then turn off the **Analysis builder UI**, your query isn't saved.

> **Tip**
>
> If a scheduled maintenance occurs while a query is running, the query is terminated and rolled back. You must restart the query.

The following topics explain how to use the analysis builder.

Topics

- Use the analysis builder to query a single table (aggregation)
- Use the analysis builder to query two tables (aggregation or list)

Use the analysis builder to query a single table (aggregation)

This procedure demonstrates how to use the **Analysis builder UI** in the AWS Clean Rooms console to build a query. The query is for a collaboration that has a single table that uses the aggregation analysis rule with no JOIN required.

To use the analysis builder to query a single table

Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

1. In the left navigation pane, choose **Collaborations**.
2. Choose the collaboration that has **Your member abilities** status of **Run queries**.
3. On the **Analysis** tab, under **Tables**, view the table and its associated analysis rule type. (The analysis rule type should be the **Aggregation analysis rule**.)

   > **Note**
   >
   > If you don’t see the table you expect, it might be for the following reasons:
   >
   > - The table hasn't been associated.
   > - The table doesn't have an analysis rule configured.

4. Under the **Analysis** section, turn on **Analysis builder UI**.
5. Build a query.

   If you want to see all of the aggregation metrics, skip to step 9.

   a. For **Choose metrics**, review the aggregate metrics that have been preselected by default and remove any metric if needed.

   b. (Optional) For **Add segments – optional**, choose one or more parameters.

> **Note**
>
> **Add segments – optional** is only displayed if dimensions are specified for the table.

c. (Optional) For **Add filters – optional**, choose **Add filter**, and then choose a **Parameter**, operator, and **Value**.

   To add more filters, choose **Add another filter**.

   To remove a filter, choose **Remove**.

> **Note**
>
> ORDER BY isn't supported for aggregation queries.
> Only the AND operator is supported in filters.

d. (Optional) For **Add description – optional**, enter a description to help identify the query in the list of queries.

6. Expand **Preview SQL code**.

   a. View the SQL code that's generated from the analysis builder.

   b. To copy the SQL code, choose **Copy**.

   c. To edit the SQL code, choose **Edit in SQL code editor**.

7. (Optional) For **Compute payer**, select the collaboration member who pays for query compute costs.

> **Note**
>
> If there is only one payer candidate for query compute in the collaboration, it defaults to that payer.

8. Choose **Run**.

> **Note**
>
> You can't run the query if the member who can receive results hasn't configured the query results settings.

9. Continue to adjust parameters and run your query again, or choose the **+** button to start a new query in a new tab.

> **Note**
>
> AWS Clean Rooms aims to provide clear error messaging. If an error message doesn't have enough details to help you troubleshoot, contact the account team. Provide them with a description of how the error occurred and the error message (including any identifiers). For more information, see Troubleshooting AWS Clean Rooms.

Use the analysis builder to query two tables (aggregation or list)

This procedure describes how to use the analysis builder in the AWS Clean Rooms console to build a query for a collaboration that has:

- Two tables (one from each member) that both use the aggregation analysis rule
- Two tables (one from each member) that both use the list analysis rule
- Two tables (one from each member) that both use the aggregation analysis rule and two tables (one from each member) that both use the list analysis rule

To use the analysis builder to query two tables

Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

1. In the left navigation pane, choose **Collaborations**.
2. Choose the collaboration that has **Your member abilities** status of **Run queries**..
3. On the **Analysis** tab, under **Tables**, view the two tables and their associated analysis rule type (**Aggregation analysis rule** or **List analysis rule**).

> **Note**
>
> If you don't see the tables you expect in the list, it might be for the following reasons:
>
> - The tables haven't been associated.
> - The tables don't have an analysis rule configured.

4. Under the **Analysis** section, turn on **Analysis builder UI**.
5. Build a query.

   If the collaboration contains two tables that use the **Aggregation analysis rule** and two tables that use the **List analysis rule**, first choose **Aggregation** or **List**, and then follow the prompts based on the selected analysis rule.

If the two tables use the aggregation analysis rule	If the two tables use the list analysis rule
1. For **Choose metrics**, review the aggregate metrics that have been preselected by default and remove any metric if needed. 2. For **Match records**, choose one or more records. **Note** When using the analysis builder, you can match only on a single pair of columns. 3. (Optional) For **Add segments – optional**, choose one or more parameters.	1. For **Choose attributes**, review the list attributes that have been preselected by default and remove any metric if needed. 2. For **Match records**, choose one or more records. **Note** When using the analysis builder, you can match only on a single pair of columns. 3. (Optional) For **Add filters – optional**, choose **Add filter**, and then choose a parameter, operator, and value. To add more filters, choose **Add another filter**. To remove a filter, choose **Remove**.

If the two tables use the aggregation analysis rule	If the two tables use the list analysis rule
Note **Add segments – optional** is only displayed if dimensions are specified for the table. 4. (Optional) For **Add filters – optional**, choose **Add filter**, and then choose a parameter, operator, and value. To add more filters, choose **Add another filter**. To remove a filter, choose **Remove**. **Note** ORDER BY is not supported for aggregation queries. Only the AND operator is supported in filters. 5. (Optional) For **Add description – optional**, enter a description to help identify the query in the list of recent queries.	**Note** LIMIT is not supported for list queries. Only the AND operator is supported in filters. 4. (Optional) For **Add description – optional**, enter a description to help identify the query in the list of recent queries.

6. Expand **Preview SQL code**.
   a. View the SQL code that's generated from the analysis builder.
   b. To copy the SQL code, choose **Copy**.
   c. To edit the SQL code, choose **Edit in SQL code editor**.
7. (Optional) For **Compute payer**, select the collaboration member who pays for query compute costs.

> **Note**
>
> If there is only one payer candidate for query compute in the collaboration, it defaults to that payer.

8. Choose **Run**.

> **Note**
>
> You can't run the query if the member who can receive results hasn’t configured the query results settings

9. Continue to adjust parameters and run your query again, or choose the **+** button to start a new query in a new tab.

> **Note**
>
> AWS Clean Rooms aims to provide clear error messaging. If an error message doesn't have enough details to help you troubleshoot, contact the account team. Provide them with a description of how the error occurred and the error message (including any identifiers). For more information, see Troubleshooting AWS Clean Rooms.

Viewing the impact of differential privacy

In general, writing and running queries doesn't change when differential privacy is turned on. However, you can't run a query if there isn't enough privacy budget remaining. As you run queries and consume the privacy budget, you can see approximately how many aggregations you can run and how that might impact future queries.

To view the impact of differential privacy in a collaboration

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member details** status of **Run queries**.

4. On the **Analysis** tab, under **Tables**, view the remaining privacy budget. This is displayed as the estimated number of **aggregation functions remaining** and the **Utility used** (rendered as a percentage).

> **Note**
>
> The estimated number of **aggregate functions remaining** and the percentage of the **Utility used** only display for the member who can query.

5. Choose **View impact** to view how much noise is injected into the results and approximately how many aggregation functions you can run.

Viewing recent queries

You can view the queries that ran in the last 90 days on the **Analysis** tab.

> **Note**
>
> If your only member ability is **Contribute data**, and you aren't the member paying for query compute costs, the **Analysis** tab doesn't appear on the console.

To view recent queries

Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

1. In the left navigation pane, choose **Collaborations**.
2. Choose a collaboration.
3. On the **Analysis** tab, under **Analyses**, select **All queries** from the dropdown, and view the queries that have been run in the last 90 days.
4. To sort recent queries by **Status**, select a status from the **All statuses** dropdown list.

   The statuses are: **Submitted**, **Started**, **Cancelled**, **Success**, **Failed**, and **Timed out**.

Viewing query details

You can view the query details as the member who can run queries or as a member who can receive results.

To view the details of the query

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose a collaboration.
4. On the **Analysis** tab, do one of the following:
   - Choose the option button for the specific query you want to view, and then choose **View details**.
   - Choose the **Protected query ID**.
5. On the **Query details** page,
   - If you are the member who can run queries, view the **Query details**, **SQL text** and **Results**.

     You see a message confirming that the query results were delivered to the member who can receive results.
   - If you are the member who can receive results, view the **Query details** and **Results**.

Exporting query analysis logs

AWS Clean Rooms runs SQL queries on Apache Spark. When a query fails or runs slowly, the Spark logs for that query contain the execution detail you need to find the cause. They show which stage failed, how work was distributed across tasks, how much memory was used, and where time was spent.

A query runs across data contributed by several collaboration members. Raw Spark logs can therefore reveal information about another member's data, such as table names, storage locations, and data values. For that reason, AWS Clean Rooms does not give you the raw logs. Instead, it produces a *redacted* copy in which customer data and member metadata are redacted. AWS Clean Rooms exports that copy to an Amazon S3 bucket that you own. You then analyze the exported logs with the tooling of your choice, such as Spark History Server.

> **Important**
>
> Exported logs are redacted. Data values are removed, and counts and sizes are reported as approximations so that they do not reveal details about the data in the collaboration. As a result, the exported logs do not match the logs that a Spark job produces outside of AWS Clean Rooms. Before you interpret them, read Understanding redacted logs.

Prerequisites

Before you can export logs for a query, all of the following must be true:

- Your membership has the `CAN_EXPORT_QUERY_ANALYSIS_LOG` member ability. Every collaboration member must approve a change request to grant this ability. For more information, see Update member abilities.
- You are the query runner or the payer of the query. Having the ability is not sufficient on its own: you can export logs only for queries that you ran or that you paid for.
- The query has reached a terminal state. You can export logs for a query with a status of `SUCCESS`, `FAILED`, `CANCELLED`, or `TIMED_OUT`. You can't export logs while a query is still running.

Log export isn't available for every query. For the cases that aren't supported, see Considerations and limitations.

You don't need to create an IAM role for log export. AWS Clean Rooms writes the exported logs using your own identity, so it writes the logs only where your own permissions already allow. As a least-privilege practice, grant write access only to the bucket and key prefix that you use for exported logs.

> **Note**
>
> If the destination bucket uses SSE-KMS with a customer managed key, you must have permission to use that key. If you don't, the export request fails immediately.

Exporting logs for a query

Log export is asynchronous. When you start an export, AWS Clean Rooms returns immediately with an export ID and a status of `IN_PROGRESS`, then redacts and copies the logs in the background.

Before starting the export, AWS Clean Rooms writes a zero-byte object named `validationSuccess` to your destination bucket to confirm that it can write there. If this check fails, the request fails immediately with a validation error instead of failing later in the background.

Console

To export logs for a query (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console.
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that contains the query.
4. Choose the **Analysis** tab.
5. Do either of the following:
   - In the table of queries, select the query, choose **Actions**, and then choose **Export query analysis logs**.
   - Choose the query to open its details page, and then choose **Export analysis logs**.
6. In the dialog box, choose the destination Amazon S3 bucket and, optionally, a key prefix.
7. Choose **Export**.

If the **Export query analysis logs** option is unavailable, the console explains why. The following table lists the messages and how to resolve them.

Message	Resolution
Query must reach a terminal state before logs can be exported.	Wait for the query to finish running.
You don't have the ability to export query analysis logs.	Ask a collaboration member to submit a change request adding the ability to your membership.

Message	Resolution
Log export is not supported for queries that use differential privacy.	None. Log export isn't available for these queries.
Only the query runner or the payer of the query can export analysis logs.	Ask the member who ran or paid for the query to export the logs.

The query details page includes an **Export history** table listing each export and its status. Choose a failed export to see the error, or choose **View destination** to open the exported logs in Amazon S3.

API

To export logs for a query (API)

Call `StartAnalysisLogExport`, specifying the membership, the protected query, and your destination.

```
aws cleanrooms start-analysis-log-export \
    --membership-identifier membership-id \
    --analysis-id protected-query-id \
    --analysis-type PROTECTED_QUERY \
    --result-configuration '{
        "outputConfiguration": {
            "s3": {
                "bucket": "amzn-s3-demo-bucket",
                "keyPrefix": "query-logs/"
            }
        }
    }'
```

The response returns an `analysisLogExport` object containing an `analysisLogExportId` and a `status` of `IN_PROGRESS`.

To check on an export, call `GetAnalysisLogExport` with the membership identifier and the export identifier, and poll until the status is `SUCCESS` or `FAILED`. When an export fails, the response includes an `error` field with a code and a message.

```
aws cleanrooms get-analysis-log-export \
```

```
    --membership-identifier membership-id \
    --analysis-log-export-identifier analysis-log-export-id
```

To list the exports for a membership, call `ListAnalysisLogExports`. You can filter the results by query using `analysisIdentifier`, and by the status of the export using `status`. The `status` filter accepts `IN_PROGRESS`, `SUCCESS`, or `FAILED`, and takes one value at a time. These are the statuses of the export itself, not of the query whose logs you exported. The results are paginated.

```
aws cleanrooms list-analysis-log-exports \
    --membership-identifier membership-id \
    --analysis-identifier protected-query-id \
    --status SUCCESS
```

Only one export can be active at a time for a given query and destination. If you start an export while another export for the same query and the same Amazon S3 destination is still in progress, the request fails with a validation error. To export the same query twice at once, use a different key prefix for the second export.

You export independently of other members. If you and another member both have the ability and both ran or paid for the same query, each of you can export the logs to your own destination.

Where the exported logs are written

AWS Clean Rooms writes the exported logs under the bucket and key prefix that you specify, using the following structure:

```
s3://your-bucket/key-prefix/collaboration=collaboration-id/analysis=protected-query-
id/analysis-log-export-id/
```

Because the path includes the export ID, exporting the same query more than once doesn't overwrite the logs from an earlier export.

An export contains the redacted Spark event log for the query. This log is a set of structured JSON records covering jobs, stages, tasks, timings, and query plans. Spark itself writes this event log format. You can process an export with any tool that reads Spark event logs, including your own tooling. AWS Clean Rooms doesn't provide a viewer for exported logs. Within the export directory, the records follow the layout that Spark uses for a rolling event log. A directory named `eventlog_v2_`*`protected-query-id`* holds one or more `events_` files and an `appstatus_` file

that records that the application completed. The directory and file names use the protected query ID rather than an internal AWS Clean Rooms identifier.

Spark History Server is one such tool. To view an export in Spark History Server, point an instance at the `eventlog_v2_` directory for the query. For example, an export of query `6ba7b810-9dad-11d1-80b4-00c04fd430c8` to the bucket `amzn-s3-demo-bucket` with the key prefix `query-logs/` produces the following.

```
s3://amzn-s3-demo-bucket/query-logs/
  collaboration=f47ac10b-58cc-4372-a567-0e02b2c3d479/
    analysis=6ba7b810-9dad-11d1-80b4-00c04fd430c8/
      a1b2c3d4-e5f6-7890-abcd-ef1234567890/
        eventlog_v2_6ba7b810-9dad-11d1-80b4-00c04fd430c8/
          events_1_6ba7b810-9dad-11d1-80b4-00c04fd430c8
          appstatus_6ba7b810-9dad-11d1-80b4-00c04fd430c8
```

In this example, set the Spark History Server log directory to the following. Pointing at the `eventlog_v2_` directory renders only the query that you exported. Reading the logs directly from Amazon S3 requires the S3A connector, so use the `s3a://` scheme. The credentials that Spark History Server runs with need read access to the bucket.

```
spark.history.fs.logDirectory s3a://amzn-s3-demo-bucket/
query-logs/collaboration=f47ac10b-58cc-4372-a567-0e02b2c3d479/
analysis=6ba7b810-9dad-11d1-80b4-00c04fd430c8/a1b2c3d4-e5f6-7890-abcd-ef1234567890/
eventlog_v2_6ba7b810-9dad-11d1-80b4-00c04fd430c8/
```

Use Spark History Server version 3.5.0 or later. You are responsible for setting up and running your own instance. For information about running a Spark History Server instance, see Viewing after the fact on the Apache Spark website.

Understanding redacted logs

AWS Clean Rooms redacts Spark logs before exporting them so that the exported logs don't reveal the contents of a member's data, or identify the table and storage locations of the source data. AWS Clean Rooms reconstructs the log records and emits only the fields that are known to be safe to share. Some information is preserved deliberately, including the names of columns that a query read.

Redaction doesn't distinguish between members. Expect your own table names and query values to be redacted from the logs that you export for your own queries, on the same terms as another

member's information. Empty and placeholder fields are normal and don't indicate a problem with the export.

What the exported logs contain

The exported logs preserve the information needed to diagnose failures and performance problems:

- **Timings** – Task and stage durations, run time, CPU time, garbage collection time, deserialization and result serialization time, shuffle wait time, and shuffle write time are all preserved exactly. Timings are the main signal for performance analysis.
- **Spark execution identifiers and counts** – Job, stage, task, and attempt IDs, partition counts, task counts, executor IDs, host names, and ports.
- **Query plan structure** – The physical plan tree, including operator names such as `HashAggregate`, `SortMergeJoin`, and `Exchange`, so you can see the shape of the plan and how the engine chose to execute your query. An operator name that isn't one of Spark's own appears as `[REDACTED]`.
- **Source column names** – Column names that come from a table being scanned, so you can see which columns are used in filters, joins, and aggregations. Names that a query creates, such as an alias, are not preserved. For more information, see How column and table names are treated.
- **Scan type and file format** – A scan operator ordinarily includes the name of the table being read. The name is replaced with a form that identifies only the scan type and file format, such as `Scan parquet`, so you can still tell how data is being read. If the file format isn't one that AWS Clean Rooms recognizes, the operator appears as `Scan` with no format.
- **Error classes** – When a query fails, the logs identify the kind of error but not the data that caused it. They report Spark's error class, its SQLSTATE code, and the standard message that Spark defines for that class, with the message's `<placeholder>` tokens left unfilled, because the values that would fill them come from the data. A failed cast, for example, reports `[CAST_INVALID_INPUT]` and a message describing that a value of one type can't be cast to another, without showing the value. When one failure caused another, the error classes in the chain are reported together, up to a few levels deep. A failure that didn't originate in Spark is reported only as a non-Spark error, with no class.

  Individual task failures report only the kind of failure, such as `ExceptionFailure`. They don't carry an error class of their own.
- **Spark configuration** – Settings whose values are always numbers, sizes, or fixed keywords, and that are most often needed to diagnose a performance problem: driver and executor cores

and memory, memory fractions and off-heap settings, dynamic allocation settings, adaptive query execution settings, default parallelism and the shuffle partition count, the broadcast join threshold, the maximum partition size, shuffle compression settings, the scheduler mode, and the serializer. Most other configuration is redacted, because settings can contain paths, identifiers, and query text. One of these settings is also omitted if its value isn't a plain number, size, or keyword.

- **Memory utilization and data volumes** – Peak execution memory, memory and disk spill, executor JVM memory, cached data sizes, shuffle block counts, and the number of bytes and records read and written, including shuffle read and write volumes. These figures are rounded down rather than exact. For more information, see How redacted logs differ from standard Spark logs.
- **Why an executor stopped** – A category for each executor that stopped, such as running out of memory, being shut down by the driver, being decommissioned, or failing to respond. This is often the fastest way to explain why a query failed. The logs also record any executor or host that Spark stopped scheduling work on, which is useful when failures keep recurring in one place. For more information, see How redacted logs differ from standard Spark logs.
- **Shuffle fetch failure detail** – When a task can't fetch shuffle output, the shuffle, map, map index, and reduce identifiers are preserved, along with the executor and host that the fetch was attempted against. This lets you distinguish repeated failures against a single source, which usually indicate an unresponsive executor, from failures spread across many sources, which usually indicate transient network problems.

What is redacted

- **Data values** – Literal values from your query, and data values from any table. A filter predicate shows which column was filtered but not the value it was compared against.
- **Table names and storage locations** – Table identifiers, database names, and Amazon S3 paths.
- **Query text** – The SQL statement and any description associated with the query.
- **Error message text and stack traces** – Because an error message can quote the value that caused the error, message text and stack traces are removed. The name of the underlying exception class is also redacted, because an exception thrown by a user-defined function carries a name that the query author chose. The Spark error class is preserved instead, for the query and the job. The reason a stage failed and the reason a task was killed are redacted as well.
- **Names that Spark records for its own work** – The names and call sites of stages and cached datasets, and the application name. These are free-text fields that can echo a table name or

a storage path. In Spark History Server, stages therefore appear without the descriptions that would normally identify them, and you locate a stage by its ID and its place in the plan instead.

- **Computed and aliased column names** – Names that a query creates rather than reads from a table, because the query engine generates such a name from the expression and the name can contain a value. These appear as a numeric identifier such as #42. For more information, see How column and table names are treated.
- **Scan detail** – The read schema, the filter predicates applied at each scan (pushed-down filters, partition filters, and data filters), and the data location.
- **Links to driver and executor logs** – The log URLs that Spark records for each executor and for the driver. In Spark History Server, the links that would normally open a driver or executor log aren't populated.
- **Resource identifiers and service endpoints** – AWS account IDs, ARNs such as those of IAM roles and AWS KMS keys, and the endpoint URLs of the services that AWS Clean Rooms calls to run your query.
- **AWS service details** – AWS Clean Rooms class names, classpath entries, JVM configuration, and Hadoop and system properties. Job properties are redacted as well, apart from the identifiers that Spark History Server needs in order to associate jobs with the query execution they belong to.

How column and table names are treated

The exported logs keep the names of columns that a query read. Table names and column aliases created in the query are redacted. A name that a query creates is generated from the expression it names, and that generated name can contain a value from the query: `SELECT 'confidential'` produces a column named `confidential`. A name therefore appears only when it can be traced back to a column of a scanned table. Every other name is replaced with a numeric identifier.

For example, consider the following query.

```
SELECT user_id, SUM(amount) AS total
FROM   sales
WHERE  region = 'us-west'
GROUP BY user_id
```

The exported logs represent it as follows.

Element of the query	In the exported logs
The columns `user_id`, `amount`, and `region`	Present by name, so you can see what the query grouped, summed, and filtered on
The table `sales`, its database, and its storage location	Not present
The value `'us-west'`	Replaced with `[REDACTED]`
The alias `total`	Replaced with a numeric identifier, because the query created the name rather than reading it from a table

How redacted logs differ from standard Spark logs

Row counts and data volumes are approximate

Record counts, byte volumes, spill sizes, and peak memory measurements are rounded down to an order of magnitude, because an exact count can reveal the size of another member's data. A task that read 1,342 records reports 1,000. Measurements below 100 report as 0.

Because every figure is rounded down, a task that reports 100 and a task that reports 1,000 might have read nearly the same number of records. Treat figures that differ by only one order of magnitude with caution; larger differences, such as 1,000 against 10,000,000, still reliably indicate skew.

Redacted values take more than one form

In the query plan, a redacted value appears as the literal text `[REDACTED]`. Elsewhere in the logs, a redacted field is usually empty instead. It might be a blank string, an empty list, or a value that is absent altogether, so that Spark History Server can still read the record. Both forms mean the same thing.

Query plans don't use familiar operator syntax

Expressions in the plan are written as function calls rather than in mathematical notation. A comparison appears as `EqualTo(#12, [REDACTED])` rather than `(a = 5)`. Function names appear in this form too, and user-defined function names don't appear at all.

Only the physical plan is included

The exported logs contain the physical plan. They don't contain the parsed, analyzed, or optimized logical plans that ordinarily accompany it.

Per-task metrics are redacted

The plan graph lists the metrics that each operator reports, such as the number of output rows. Metrics that individual tasks report show no value at all, because the per-task figures behind them are removed. For task-level numbers, use the per-task metrics in the stage view instead.

Spark driver metrics are rounded down

Metrics that Spark computes on the driver, including those for broadcasts, scans, and writes, do report a value, but it is rounded down to an order of magnitude for the same reason that record counts and data volumes are: an exact figure can reveal the size of another member's data.

Only the event log is exported

The export contains the Spark event log. It doesn't contain the free-form driver and executor output that Spark writes alongside it, because a single line of that output can contain a table name, a storage path, or a data value. Diagnosis has to be done from the event log.

Executor failures show generic error messages

Because the free-form reason an executor stopped can contain AWS Clean Rooms service detail, the logs instead report a general error message, such as an out-of-memory exit, a shutdown initiated by the driver, a decommissioned executor, or a lost process. This record is specific to AWS Clean Rooms and isn't a standard Spark event, so Spark History Server and other tools might not display it. If you don't see it, you can find it in the exported event log file.

Unrecognized records are redacted by default

If a log record isn't one that AWS Clean Rooms recognizes, all of its text is replaced with `[REDACTED]`, including its field names, and all of its numbers are replaced with zero. Only the record type is preserved. Such a record carries no diagnostic information.

Troubleshooting log export

The request fails because you don't have permission to export logs

Your membership doesn't have the `CAN_EXPORT_QUERY_ANALYSIS_LOG` ability. A collaboration member must submit a change request to add the ability, and all members must approve it. For more information, see Update member abilities.

The request fails because the bucket can't be written to

Verify that the bucket exists, that it's in the same AWS Region as the collaboration, and that you have permission to write to it. If the bucket policy denies `s3:PutObject`, or restricts writes to a specific VPC endpoint, AWS Clean Rooms can't write to the bucket.

If the bucket uses SSE-KMS with a customer managed key, this error might also mean that you don't have permission to use that key. Confirm that the key policy allows you to call `kms:GenerateDataKey` and `kms:Decrypt` through Amazon S3.

The export fails because the query never ran

The query reached a terminal state without running on Spark, so no logs were produced. This happens when a query fails validation or is cancelled before it starts. The export fails with an error code of `LOGS_NOT_AVAILABLE` and a message reporting that no query execution logs are available for the query. There is nothing to export.

The request fails because the query ran before log export was available

Redacted logs are produced while a query runs. A query that ran before log export became available in AWS Clean Rooms produced none, so there is nothing to export. To get logs for the same analysis, run the query again and export the logs for the new run.

The request fails because a quota was exceeded

You have reached the maximum number of exports that can be in progress at the same time. Wait for an in-progress export to finish, then retry.

The export starts but then fails

The initial permission check confirms only that AWS Clean Rooms can write to your bucket at the moment you start the export. If your permissions change, or if the credentials used for the export are revoked while the export is running, the export fails after it has started. Call `GetAnalysisLogExport` to see the error, resolve the cause, and start a new export.

The export fails and reports an internal error

An internal error might mean that no logs were produced for the query. You can retry the export.

The exported logs appear incomplete

If the query ended because a process ran out of memory, some log output might be missing, because logs held on a host that stops abruptly aren't recovered. The logs that were already written are still exported.

There is a zero-byte `validationSuccess` object in the bucket

AWS Clean Rooms writes this object at the key prefix that you specified, to verify that it can write to your destination before starting an export. It's separate from the exported logs, which are written under the export's own directory, and you can delete it. End your key prefix with a slash so that this object is created inside the prefix rather than beside it.

Considerations and limitations

- You can export logs for SQL queries run after August 11, 2026. Log export doesn't support PySpark jobs.
- Log export isn't supported for queries that failed validation or were cancelled before reaching the `STARTED` status.
- Log export isn't supported for queries that use differential privacy. For more information, see AWS Clean Rooms Differential Privacy.
- The destination bucket for log export must be in the same AWS Region as the collaboration. Cross-Region export isn't supported.
- Securing the destination bucket for log export is your responsibility. Block public access to the bucket, and enable server access logging if you need a record of who reads the exported logs.
- AWS Clean Rooms doesn't accept a AWS KMS key for log export. Exported logs are encrypted using the default encryption configuration of the destination Amazon S3 bucket. To encrypt them with a customer managed key, configure the bucket's default encryption to use that key before you export.
- Log export supports only queries that reached a terminal state: `SUCCESS`, `FAILED`, `CANCELLED`, or `TIMED_OUT`.
- You can't cancel an export after it starts.

For quotas that apply to log export, including the number of exports that can run at the same time, see Quotas for AWS Clean Rooms.

Running PySpark jobs

As the member who can query, you can run a PySpark job on a configured table by using an approved PySpark analysis template.

Prerequisites

Before you run a PySpark job, you must have:

- An active membership in AWS Clean Rooms collaboration
- Access to at least one analysis template in the collaboration
- Access to at least one configured table in the collaboration
- Permissions to write the results of a PySpark job to a specified S3 bucket

  For information about creating the required service role, see Create a service role to write results of a PySpark job.
- The member who is responsible to pay for compute costs has joined the collaboration as an active member

For information about how to query data or view queries by calling the AWS Clean Rooms `StartProtectedJob` API operation directly or by using the AWS SDKs, see the AWS Clean Rooms API Reference.

For information about job logging, see Analysis logging in AWS Clean Rooms.

For information about receiving job results, see Receiving and using analysis results.

The following topics explain how to run a PySpark job on a configured table in a collaboration using the AWS Clean Rooms console.

Topics

- Running a PySpark job on a configured table using a PySpark analysis template
- Viewing recent jobs
- Viewing job details

Running a PySpark job on a configured table using a PySpark analysis template

This procedure demonstrates how to use a PySpark analysis template in the AWS Clean Rooms console to analyze configured tables with the **Custom** analysis rule.

To run a PySpark job on a configured table using a PySpark analysis template

Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.

1. In the left navigation pane, choose **Collaborations**.
2. Choose the collaboration that has **Your member abilities** status of **Run jobs**.
3. On the **Analysis** tab, under the **Tables** section, view the tables and their associated analysis rule type (**Custom analysis rule**).

   > **Note**
   >
   > If you don't see the tables that you expect in the list, it might be for the following reasons:
   >
   > - The tables haven't been associated.
   > - The tables don't have an analysis rule configured.

4. Under the **Analysis** section, for **Analysis mode**, select **Run analysis templates**.
5. Choose the PySpark analysis template from the **Analysis template** dropdown list.

   The parameters from the PySpark analysis template will automatically populate in the **Definition**.
6. If the analysis template has parameters defined, under **Parameters**, provide values for the parameters:

   a. For each parameter, view the **Parameter name** and **Default value** (if configured).

   b. Enter a **Value** for each parameter you want to override.

> **Note**
>
> If you don't provide a value but a default value exists, the default value will be used.

> **Important**
>
> Parameter values can be up to 1,000 characters and support UTF-8 encoding. All parameter values are treated as strings and passed to your user script through the context object.
> Ensure that your user script validates and handles parameter values safely. For more information about secure parameter handling, see Working with parameters in PySpark analysis templates.

7. Specify the supported **Worker type** and the **Number of workers**.

   Use the following table to determine the type and number or workers you need for your use case.

Worker type	vCPU	Memory (GB)	Storage (GB)	Number of workers	Total Clean Rooms Processing Units (CRPU)
CR.1X (default)	4	30	100	4	8
				128	256
CR.4X	16	120	400	4	32
				32	256

> **Note**
>
> Different worker types and number of workers have associated costs. To learn more about the pricing, see AWS Clean Rooms pricing.

8. Specify the supported **Spark properties**.

   a. Select **Add Spark properties**.

   b. On the **Spark properties** dialog box, choose a **Property name** from the dropdown list and enter a **Value**.

The following tables provide a definition for each property.

For more information about Spark properties, see Spark Properties in the Apache Spark documentation.

> **Note**
>
> You can configure a maximum of 50 Spark properties. Each property value can be up to 500 characters.

Property Name	Description	Default Value
spark.task.maxFailures	Controls how many consecutive times a task can fail before the job fails. Requires a value greater than or equal to 1. The number of allowed retries equals this value minus 1. The failure count resets if any attempt succeeds. Failures across different tasks don't accumulate toward this limit.	4
spark.sql.files.maxPartitionBytes	Sets the maximum number of bytes to pack into a single partition when reading from	128MB

Property Name	Description	Default Value
	file-based sources such as Parquet, JSON, and ORC.	
spark.hadoop.fs.s3.maxRetries	Sets the maximum number of retry attempts for Amazon S3 file operations.	(none)
spark.network.timeout	Sets the default timeout for all network interactions. Overrides the following timeout settings if they aren't configured: • spark.storage.blockManagerHeartbeatTimeoutMs • spark.shuffle.io.connectionTimeout • spark.rpc.askTimeout • spark.rpc.lookupTimeout	120s
spark.rdd.compress	Specifies whether to compress serialized RDD partitions using spark.io.compression.codec. Applies to StorageLevel.MEMORY_ONLY_SER in Java and Scala, or StorageLevel.MEMORY_ONLY in Python. Reduces storage space but requires additional CPU processing time.	false
spark.shuffle.spill.compress	Specifies whether to compress shuffle spill data using spark.io.compression.codec.	true
spark.shuffle.compress	Specifies whether to compress map output files. Compression uses spark.io.compression.codec.	true
spark.shuffle.service.index.cache.size	Sets the cache size limit, in bytes unless otherwise specified.	100m

Property Name	Description	Default Value
spark.shuffle.io.maxRetries	Sets the maximum number of retries for fetches that fail due to IO-related exceptions.	3
spark.shuffle.io.retryWait	Sets the wait time between retries of fetches. The maximum delay caused by retrying is 15 seconds by default, calculated as maxRetries * retryWait.	5s
spark.shuffle.io.connectionTimeout	Sets the timeout for established connections between shuffle servers and clients to be marked as idle and closed if there are still outstanding fetch requests but no traffic on the channel.	(value of spark.network.timeout)
spark.driver.maxResultSize	Sets the total size limit of serialized results of all partitions for each Spark action, in bytes. Should be at least 1M, or 0 for unlimited.	1g
spark.memory.fraction	Sets the fraction of (heap space - 300MB) used for execution and storage. The lower this value, the more frequently spills and cached data eviction occur. Leaving this at the default value is recommended.	0.6
spark.scheduler.mode	Sets the scheduling mode between jobs submitted to the same SparkContext. Can be set to FAIR to use fair sharing instead of queueing jobs one after another. Supported values: FAIR, FIFO.	FIFO

Property Name	Description	Default Value
spark.sql.adaptive.advisoryPartitionSizeInBytes	Sets the target size in bytes for shuffle partitions during adaptive optimization when spark.sql.adaptive.enabled is true. Controls partition size when coalescing small partitions or splitting skewed partitions.	(value of spark.sql.adaptive.shuffle.targetPostShuffleInputSize)
spark.sql.adaptive.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Applies only in adaptive framework. Uses the same default value as spark.sql.autoBroadcastJoinThreshold. Set to -1 to disable broadcasting.	(none)
spark.sql.adaptive.coalescePartitions.enabled	Specifies whether to coalesce contiguous shuffle partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes to optimize task size. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Defines the initial number of shuffle partitions before coalescing. Requires both spark.sql.adaptive.enabled and spark.sql.adaptive.coalescePartitions.enabled to be true. Defaults to the value of spark.sql.shuffle.partitions.	(none)
spark.sql.adaptive.coalescePartitions.minPartitionSize	Sets the minimum size for coalesced shuffle partitions to prevent partitions from becoming too small during adaptive optimization.	1 MB

Property Name	Description	Default Value
spark.sql.adaptive.coalescePartitions.parallelismFirst	Specifies whether to calculate partition sizes based on cluster parallelism instead of spark.sql.adaptive.advisoryPartitionSizeInBytes during partition coalescing. Generates smaller partition sizes than the configured target size to maximize parallelism. We recommend setting this to false on busy clusters to improve resource utilization by preventing excessive small tasks.	true
spark.sql.adaptive.enabled	Specifies whether to enable adaptive query execution to re-optimize query plans during query execution, based on accurate runtime statistics.	true
spark.sql.adaptive.forceOptimizeSkewedJoin	Specifies whether to force enable OptimizeSkewedJoin even if it introduces extra shuffle.	false
spark.sql.adaptive.localShuffleReader.enabled	Specifies whether to use local shuffle readers when shuffle partitioning isn't required, such as after converting from sort-merge joins to broadcast-hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Sets the maximum partition size in bytes for building local hash maps. Prioritizes shuffled hash joins over sort-merge joins when: • This value equals or exceeds spark.sql.adaptive.advisoryPartitionSizeInBytes • All partition sizes are within this limit Overrides spark.sql.join.preferSortMergeJoin setting.	0 bytes
spark.sql.adaptive.optimizeSkewsInRebalancePartitions.enabled	Specifies whether to optimize skewed shuffle partitions by splitting them into smaller partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.rebalancePartitionsSmallPartitionFactor	Defines the size threshold factor for merging partitions during splitting. Partitions smaller than this factor multiplied by spark.sql.adaptive.advisoryPartitionSizeInBytes are merged.	0.2
spark.sql.adaptive.skewJoin.enabled	Specifies whether to handle data skew in shuffled joins by splitting and optionally replicating skewed partitions. Applies to sort-merge and shuffled hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Determines the size factor that determines partition skew. A partition is skewed when its size exceeds both: • This factor multiplied by the median partition size • The value of spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Sets the size threshold in bytes for identifying skewed partitions. A partition is skewed when its size exceeds both: • This threshold • The median partition size multiplied by spark.sql.adaptive.skewJoin.skewedPartitionFactor We recommend setting this value larger than spark.sql.adaptive.advisoryPartitionSizeInBytes.	256MB
spark.sql.broadcastTimeout	Controls the timeout period in seconds for the broadcast operations during broadcast joins.	300 seconds
spark.sql.cbo.enabled	Specifies whether to enable cost-based optimization (CBO) for plan statistics estimation.	false
spark.sql.cbo.joinReorder.dp.star.filter	Specifies whether to apply star-join filter heuristics during cost-based join enumeration.	false

Property Name	Description	Default Value
spark.sql.cbo.joinReorder.dp.threshold	Sets the maximum number of joined nodes allowed in the dynamic programming algorithm.	12
spark.sql.cbo.joinReorder.enabled	Specifies whether to enable join reordering in cost-based optimization (CBO).	false
spark.sql.cbo.planStats.enabled	Specifies whether to fetch row counts and column statistics from the catalog during logical plan generation.	false
spark.sql.cbo.starSchemaDetection	Specifies whether to enable join reordering based on star schema detection.	false
spark.sql.files.maxPartitionNum	Sets the target maximum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Rescales partitions when the initial count exceeds this value. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.files.maxRecordsPerFile	Sets the maximum number of records to write to a single file. No limit applies when set to zero or a negative value.	0
spark.sql.files.minPartitionNum	Sets the target minimum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Defaults to spark.sql.leafNodeDefaultParallelism. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.inMemoryColumnarStorage.batchSize	Controls the batch size for columnar caching. Increasing the size improves memory utilization and compression but increases the risk of out-of-memory errors.	10000

Property Name	Description	Default Value
spark.sql.inMemoryColumnarStorage.compressed	Specifies whether to automatically select compression codecs for columns based on data statistics.	true
spark.sql.inMemoryColumnarStorage.enableVectorizedReader	Specifies whether to enable vectorized reading for columnar caching.	true
spark.sql.legacy.allowHashOnMapType	Specifies whether to allow hash operations on map type data structures. This legacy setting maintains compatibility with older Spark versions' map type handling.	(none)
spark.sql.legacy.allowNegativeScaleOfDecimal	Specifies whether to allow negative scale values in decimal type definitions. This legacy setting maintains compatibility with older Spark versions that supported negative decimal scales.	(none)
spark.sql.legacy.castComplexTypesToString.enabled	Specifies whether to enable legacy behavior for casting complex types to strings. Maintains compatibility with older Spark versions' type conversion rules.	(none)
spark.sql.legacy.charVarcharAsString	Specifies whether to treat CHAR and VARCHAR types as STRING types. This legacy setting provides compatibility with older Spark versions' string type handling.	(none)
spark.sql.legacy.createEmptyCollectionUsingStringType	Specifies whether to create empty collections using string type elements. This legacy setting maintains compatibility with older Spark versions' collection initialization behavior.	(none)

Property Name	Description	Default Value
spark.sql.legacy.exponentLiteralAsDecimal.enabled	Specifies whether to interpret exponential literals as decimal types. This legacy setting maintains compatibility with older Spark versions' numeric literal handling.	(none)
spark.sql.legacy.json.allowEmptyString.enabled	Specifies whether to allow empty strings in JSON processing. This legacy setting maintains compatibility with older Spark versions' JSON parsing behavior.	(none)
spark.sql.legacy.parquet.int96RebaseModeInRead	Specifies whether to use legacy INT96 timestamp rebase mode when reading Parquet files. This legacy setting maintains compatibility with older Spark versions' timestamp handling.	(none)
spark.sql.legacy.timeParserPolicy	Controls the time parsing behavior for backwards compatibility. This legacy setting determines how timestamps and dates are parsed from strings.	(none)
spark.sql.legacy.typeCoercion.datetimeToString.enabled	Specifies whether to enable legacy type coercion behavior when converting datetime values to strings. Maintains compatibility with older Spark versions' datetime conversion rules.	(none)
spark.sql.maxSinglePartitionBytes	Sets the maximum partition size in bytes. The planner introduces shuffle operations for larger partitions to improve parallelism.	128m

Property Name	Description	Default Value
spark.sql.metadataCacheTTLSeconds	Controls the time-to-live (TTL) for metadata caches. Applies to partition file metadata and session catalog caches. Requires: • A positive value greater than zero • spark.sql.catalogImplementation set to hive • spark.sql.hive.filesourcePartitionFileCacheSize greater than zero • spark.sql.hive.manageFilesourcePartitions set to true	-1000ms
spark.sql.optimizer.collapseProjectAlwaysInline	Specifies whether to collapse adjacent projections and inline expressions, even when it causes duplication.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	Specifies whether to generate predicates for partition columns used as join keys.	true
spark.sql.optimizer.enableCsvExpressionOptimization	Specifies whether to optimize CSV expressions in SQL optimizer by pruning unnecessary columns from from_csv operations.	true
spark.sql.optimizer.enableJsonExpressionOptimization	Specifies whether to optimize JSON expressions in SQL optimizer by: • Pruning unnecessary columns from from_json operations • Simplifying from_json and to_json combinations • Optimizing named_struct operations	true

Property Name	Description	Default Value
spark.sql.optimizer.excludedRules	Defines optimizer rules to disable, identified by comma-separated rule names. Some rules cannot be disabled as they are required for correctness. The optimizer logs which rules are successfully disabled.	(none)
spark.sql.optimizer.runtime.bloomFilter.applicationSideScanSizeThreshold	Sets the minimum aggregated scan size in bytes required to inject a Bloom filter on the application side.	10GB
spark.sql.optimizer.runtime.bloomFilter.creationSideThreshold	Defines the maximum size threshold for injecting a Bloom filter on the creation side.	10MB
spark.sql.optimizer.runtime.bloomFilter.enabled	Specifies whether to insert a Bloom filter to reduce shuffle data when one side of a shuffle join has a selective predicate.	true
spark.sql.optimizer.runtime.bloomFilter.expectedNumItems	Defines the default number of expected items in the runtime Bloom filter.	1000000
spark.sql.optimizer.runtime.bloomFilter.maxNumBits	Sets the maximum number of bits allowed in the runtime Bloom filter.	67108864
spark.sql.optimizer.runtime.bloomFilter.maxNumItems	Sets the maximum number of expected items allowed in the runtime Bloom filter.	4000000
spark.sql.optimizer.runtime.bloomFilter.numBits	Defines the default number of bits used in the runtime Bloom filter.	8388608

Property Name	Description	Default Value
spark.sql.optimizer.runtime.rowLevelOperationGroupFilter.enabled	Specifies whether to enable runtime group filtering for row-level operations. Allows data sources to: • Prune entire groups of data (such as files or partitions) using data source filters • Execute runtime queries to identify matching records • Discard unnecessary groups to avoid expensive rewrites Limitations: • Not all expressions can convert to data source filters • Some expressions require Spark evaluation (such as subqueries)	true
spark.sql.optimizer.runtimeFilter.number.threshold	Sets the total number of injected runtime filters (non-DPP). This is to prevent driver OOMs with too many Bloom filters.	10
spark.sql.optimizer.runtimeFilter.semiJoinReduction.enabled	Specifies whether to insert a semi-join to reduce shuffle data when one side of a shuffle join has a selective predicate.	false

Property Name	Description	Default Value
spark.sql.parquet.aggregatePushdown	Specifies whether to push down aggregates to Parquet for optimization. Supports: • MIN and MAX for boolean, integer, float, and date types • COUNT for all data types Throws an exception if statistics are missing from any Parquet file footer.	false
spark.sql.parquet.columnarReaderBatchSize	Controls the number of rows in each Parquet vectorized reader batch. Choose a value that balances performance overhead and memory usage to prevent out-of-memory errors.	4096
spark.sql.parquet.enableVectorizedReader	Specifies whether to enable vectorized Parquet decoding.	true
spark.sql.shuffle.partitions	Sets the default number of partitions for data shuffling during joins or aggregations. Cannot be modified between structured streaming query restarts from the same checkpoint location.	200
spark.sql.shuffledHashJoinFactor	Defines the multiplication factor used to determine shuffle hash join eligibility. A shuffle hash join is selected when the small-side data size multiplied by this factor is less than the large-side data size.	3

Property Name	Description	Default Value
spark.sql.sources.parallelPartitionDiscovery.threshold	Sets the maximum number of paths for driver-side file listing with file-based sources (Parquet, JSON, and ORC). When exceeded during partition discovery, files are listed using a separate Spark distributed job.	32
spark.sql.statistics.histogram.enabled	Specifies whether to generate equi-height histograms during column statistics computation to improve estimation accuracy. Requires an additional table scan beyond the one needed for basic column statistics.	false
spark.dynamicAllocation.executorIdleTimeout	Sets the duration an executor must be idle before it is removed when dynamic allocation is enabled.	60s
spark.dynamicAllocation.schedulerBacklogTimeout	Sets the duration that pending tasks must be backlogged before new executors are requested when dynamic allocation is enabled.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Same as spark.dynamicAllocation.schedulerBacklogTimeout, but used only for subsequent executor requests.	(value of spark.dynamicAllocation.schedulerBacklogTimeout)

Property Name	Description	Default Value
spark.scheduler.minRegisteredResourcesRatio	Sets the minimum ratio of registered resources (registered resources / total expected resources) to wait for before scheduling begins. Specified as a double between 0.0 and 1.0. Regardless of whether the minimum ratio of resources has been reached, the maximum amount of time it will wait before scheduling begins is controlled by spark.scheduler.maxRegisteredResourcesWaitingTime.	0.8
spark.scheduler.maxRegisteredResourcesWaitingTime	Sets the maximum amount of time to wait for resources to register before scheduling begins.	30s
spark.sql.hive.metastorePartitionPruningFallbackOnException	Specifies whether to fall back to getting all partitions from Hive metastore and perform partition pruning on the Spark client side when encountering MetaException from the metastore.	false
spark.sql.crossJoin.enabled	Specifies whether to allow queries that contain a cartesian product without explicit CROSS JOIN syntax.	true
spark.sql.analyzer.maxIterations	Sets the maximum number of iterations the query analyzer runs before giving up. Higher values allow the analyzer to process very large or deeply nested queries.	100
spark.sql.datapreffetch.filescan.maxParallelismPerTask	Sets the maximum number of file splits to pre-fetch concurrently for each task when scanning files.	4

Property Name	Description	Default Value
spark.sql.iceberg.data-prefetch.enabled	Specifies whether to enable data pre-fetch optimization when reading Iceberg tables.	true
spark.sql.legacy.nullValueWrittenAsQuotedEmptyStringCsv	Specifies whether to restore the legacy behavior of writing nulls as quoted empty strings in CSV output. When false, Spark writes nulls as unquoted empty strings.	false
spark.maxRemoteBlockSizeFetchToMem	Sets the size threshold above which Spark fetches remote blocks to disk instead of memory. This avoids a single large request consuming too much memory.	200m
spark.emr-serverless.allocation.batch.size	Sets the number of executors to request at once in each round of executor allocation.	20

Property Name	Description	Default Value
spark.sql.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Set to -1 to disable broadcasting.	10MB
spark.io.compression.codec	Sets the codec used to compress internal data such as RDD partitions, event log, broadcast variables, and shuffle outputs. Supported values: lz4, snappy, zstd, gzip.	lz4

Property Name	Description	Default Value
spark.sql.session.timeZone	Defines the session time zone for handling timestamps in string literals and Java object conversion. Accepts: • Region-based IDs in area/city format (such as America/Los_Angeles) • Zone offsets in (+/-)HH, (+/-)HH:mm, or (+/-)HH:mm:ss format (such as -08 or +01:00) • UTC or Z as aliases for +00:00	(value of local timezone)
spark.cleanrooms.executor.memoryOverheadFactor	Sets the fraction of total executor memory used to determine the split between spark.executor.memory and spark.executor.memoryOverhead. Specified as a double between 0.0 and less than 1.0.	0.1
spark.cleanrooms.driver.memoryOverheadFactor	Sets the fraction of total driver memory used to determine the split between spark.driver.memory and spark.driver.memoryOverhead. Specified as a double between 0.0 and less than 1.0.	0.1
spark.memory.storageFraction	Sets the amount of storage memory immune to eviction, expressed as a fraction of the size of the region set aside by spark.memory.fraction. The higher this is, the less working memory may be available to execution and tasks may spill to disk more often. Leaving this at the default value is recommended.	0.5
spark.rpc.askTimeout	Sets the duration for an RPC ask operation to wait before timing out.	(value of spark.network.timeout)

Property Name	Description	Default Value
spark.executor.heartbeatInterval	Sets the interval between each executor's heartbeats to the driver. Heartbeats let the driver know that the executor is still alive and update it with metrics for in-progress tasks. spark.executor.heartbeatInterval should be significantly less than spark.network.timeout.	10s
spark.stage.maxConsecutiveAttempts	Sets the number of consecutive stage attempts allowed before a stage is aborted.	4
spark.task.cpus	Sets the number of cores to allocate for each task.	1
spark.shuffle.file.buffer	Sets the size of the in-memory buffer for each shuffle file output stream, in KiB unless otherwise specified. These buffers reduce the number of disk seeks and system calls made in creating intermediate shuffle files.	32k
spark.reducer.maxSizeInFlight	Sets the maximum size of map outputs to fetch simultaneously from each reduce task, in MiB unless otherwise specified. Since each output requires a buffer to receive it, this represents a fixed memory overhead per reduce task, so keep it small unless you have a large amount of memory.	48m

9. (Optional) For **Compute payer**, select the collaboration member who pays for job compute costs.

> **Note**
>
> If there is only one payer candidate for job compute in the collaboration, it defaults to that payer.

10. Choose **Run**.

> **Note**
>
> You can't run the job if the member who can receive results hasn't configured the job results settings.

11. Continue to adjust parameters and run your job again, or choose the **+** button to start a new job in a new tab.

Viewing recent jobs

You can view the jobs that ran in the last 90 days on the **Analysis** tab.

> **Note**
>
> If your only member ability is **Contribute data**, and you aren't the member paying for job compute costs, the **Analysis** tab doesn't appear on the console.

To view recent jobs

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose a collaboration.
4. On the **Analysis** tab, under **Analyses**, select **All jobs** from the dropdown, and view the jobs that have been run in the last 90 days.
5. To sort recent jobs by **Status**, select a status from the **All statuses** dropdown list.

   The statuses are: **Submitted**, **Started**, **Cancelled**, **Success**, **Failed**, and **Timed out**.

Viewing job details

You can view the job details as the member who can run jobs or as a member who can receive results.

To view the details of the job

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. Choose a collaboration.
4. On the **Analysis** tab, under **Analyses**, select **All jobs** from the dropdown, and then do one of the following:
   - Choose the option button for the specific job you want to view, and then choose **View details**.
   - Choose the **Protected job ID**.
5. On the **Job details** page,
   - If you are the member who can run jobs, view the **Job details**, **Job**, and **Results**.

     You see a message confirming that the job results were delivered to the member who can receive results.
   - If you are the member who can receive results, view the **Job details** and **Results**.

Using an intermediate table in analyses

After you populate an intermediate table and attach an analysis rule, you can reference it in queries just like a configured table.

Discovering intermediate tables

All collaboration members can view intermediate tables on the **Tables** tab of a collaboration. The table type is shown as "Intermediate table."

Referencing in SQL

Reference an intermediate table by name in your SQL query, just like you reference a configured table. For example:

```
SELECT * FROM my_intermediate_table
```

Referencing in PySpark

In a PySpark analysis template, reference an intermediate table through the context object:

```
context['referencedTables']['my_intermediate_table']
```

Requirements at analysis time

AWS Clean Rooms enforces the following requirements when you run an analysis that references an intermediate table:

- The intermediate table schema status must be **Ready** (populated and analysis rule attached).
- The analysis rule is validated against inherited constraints from base tables.
- Deferred controls are enforced, including disallowedOutputColumns and allowedResultReceivers.

Using in ML workflows

You can reference intermediate tables in ML workflows through the dataSource.protectedQueryInputParameters configuration.

Viewing the schema

To view the columns available in an intermediate table, choose the **Schema** tab on the intermediate table details page. You can view the column name and type. Choose **Edit schema** to update column types.

Viewing dependencies

To view the base tables that an intermediate table depends on, choose the **Dependencies** tab. You can view the table name, owner, type, and parent type for each dependency.

Receiving and using analysis results

Members who can receive results can review the query results in either the AWS Clean Rooms console or in the Amazon S3 buckets that they specified when they joined the collaboration.

> **Note**
>
> For encrypted data tables only, the member who can receive results decrypts the query results by running the C3R encryption client in the decrypt mode.
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

The following topics explain how to receive analysis results using the AWS Clean Rooms console.

Topics

- Receiving query results
- Receiving job results
- Editing default values for query results settings
- Editing default values for job results settings
- Using query output in other AWS services

For information about how to query data or view queries by calling the AWS Clean Rooms API directly or by using the AWS SDKs, see the AWS Clean Rooms API Reference.

For information about query logging, see Analysis logging in AWS Clean Rooms.

> **Note**
>
> If you run a query on encrypted data tables, the results from the encrypted columns are encrypted.

Receiving query results

> **Note**
>
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

The results of the query are located in the **Results settings defaults** section of the **Analysis** tab in the AWS Clean Rooms console.

To receive query results

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Receive results**.
4. To receive the query results directly from AWS Clean Rooms, on the **Analysis** tab, under **Analyses**, select **All queries** from the dropdown, and then under the **Protected query ID** column, select the query.
5. On the **Query details** page, under **Results**, do one of the following:

If you want to...	Then choose...
Copy the results.	**Copy**
Download the results.	**Download** **Note** By default, the downloaded file's name is the corresponding `Query id` that was displayed when the query was run in AWS Clean Rooms.
View the results in Amazon S3.	**View in Amazon S3**

If you want to...	Then choose...
	The Amazon S3 console opens in a separate tab.

6. If you're using encrypted data, you can now decrypt the data tables.

   For more information, see Decrypting data tables with the C3R encryption client.

Receiving job results

> **Note**
>
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

The results of the job are located in the **Results settings defaults** section of the **Analysis** tab in the AWS Clean Rooms console.

To receive job results

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Receive results**.
4. To receive the job results directly from AWS Clean Rooms, on the **Analysis** tab, under **Analyses**, select **All jobs** from the dropdown, and then under the **Protected job ID** column, select the job.
5. On the **Job details** page, under **Results**, copy the Job ID.

   Go back to the **Analysis** tab and expand the **Result settings defaults**.

   Under **Results destination**, select the link to view the results in Amazon S3.

   The Amazon S3 console opens in a separate tab.

   In Amazon S3, paste the Job ID in the Search bar and press enter.

The folder containing the results appears. Select the folder to view the job results.

Editing default values for query results settings

> **Note**
>
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

As a member who can receive results, you can edit the default values for query results settings in the AWS Clean Rooms console.

To edit the default values for query results settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Receive results**.
4. On the **Analysis** tab, under **Results settings defaults**, choose **Edit**.
5. On the **Edit results settings defaults** page, modify any of the following, as needed:

   a. Under **Query results**, modify the **Results destination in Amazon S3**, the **Result format**, or the **Result files**.

   b. (Optional) For **Service access**, if you want to deliver queries that take up to 24 hours to your S3 destination, select the **Add a service role to support queries that take up to 24 hours to complete** checkbox.

      Large queries that take up to 24 hours to complete will be delivered to your S3 destination.

      If you don't select the checkbox, only queries that complete within 12 hours will be delivered to your S3 location.

      - Specify the **Service access** permissions by selecting either **Create and use a new service role** or **Use an existing service role**.

Create and use a new service role

- AWS Clean Rooms creates a service role with the required policy for this table.
- The default **Service role name** is `cleanrooms-query-receiver-<timestamp>`
- You must have permissions to create roles and attach policies.

Use an existing service role

1. Choose an **Existing service role name** from the dropdown list.

   The list of roles are displayed if you have permissions to list roles.

   If you don't have permissions to list roles, you can enter the Amazon Resource Name (ARN) of the role that you want to use.

2. View the service role by choosing the **View in IAM** external link.

   If there are no existing service roles, the option to **Use an existing service role** is unavailable.

   By default, AWS Clean Rooms doesn't attempt to update the existing role policy to add necessary permissions.

> **Note**
>
> - AWS Clean Rooms requires permissions to query according to the analysis rules. For more information about permissions for AWS Clean Rooms, see AWS managed policies for AWS Clean Rooms.
> - If the role doesn’t have sufficient permissions for AWS Clean Rooms, you receive an error message stating that the role doesn't have sufficient permissions for AWS Clean Rooms. The role policy must be added before proceeding.
> - If you can’t modify the role policy, you receive an error message stating that AWS Clean Rooms couldn't find the policy for the service role.

6. Choose **Save changes**.

7. The updated **Query results settings** appear on the collaboration detail page.

Editing default values for job results settings

> **Note**
>
> The **Results destination in Amazon S3** can't be within the same S3 bucket as any data source.

As a member who can receive results, you can edit the default values for job results settings in the AWS Clean Rooms console.

To edit the default values for job results settings

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. Choose the collaboration that has **Your member abilities** status of **Receive results**.
4. On the **Analysis** tab, under **Results settings defaults**, choose **Edit**.
5. On the **Edit results settings defaults** page, modify any of the following, as needed:
   a. Under **Job results**, modify the **Results destination in Amazon S3**.
   b. Under **Service access**, modify the **Existing service role name**.
6. Choose **Save changes**.
7. The updated **Job results settings** appear on the collaboration detail page.

Using query output in other AWS services

SQL query output can be used for the seed data for a Clean Rooms ML model. For more information, see AWS Clean Rooms ML.

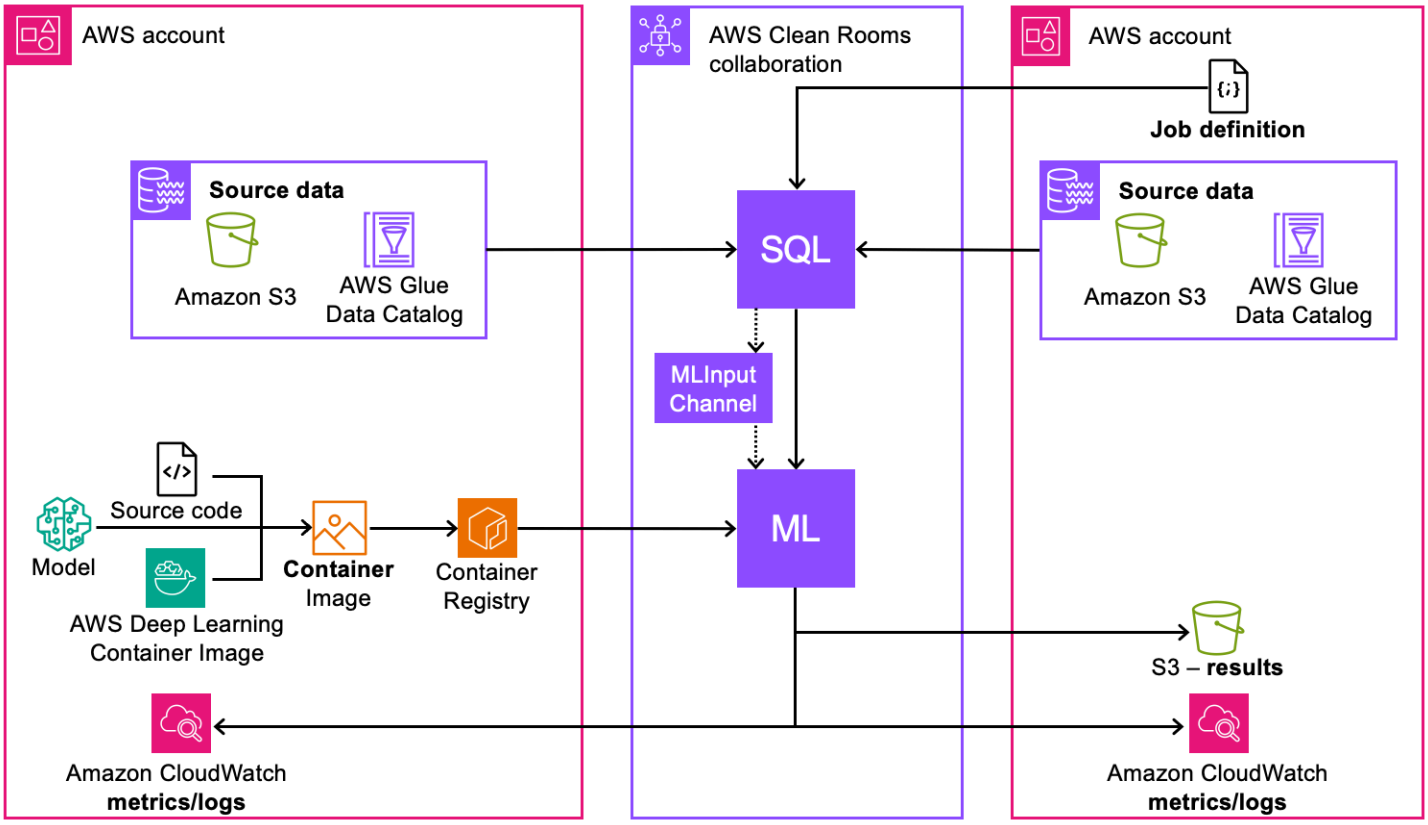
Query output from AWS Clean Rooms is available on the console (if the console is used to run queries) and downloaded in a specified Amazon S3 bucket. From there, you can use the query output in other AWS services, such as Amazon Quick and Amazon SageMaker AI, depending on how those services use data from Amazon S3.

For more information about Amazon Quick, see the Amazon Quick Documentation.

For more information about Amazon SageMaker AI, see the Amazon SageMaker AI Documentation.

Custom modeling in AWS Clean Rooms ML

From a technical standpoint, the following diagram describes how custom ML modeling works in AWS Clean Rooms ML.



Here's how custom ML modeling works in Clean Rooms ML:

1. Data Source Configuration
   - Source data can be stored in Amazon S3 catalog, in the AWS Glue Data Catalog, or Snowflake
   - AWS Glue Data Catalog is used to organize and catalog
   - Data from multiple AWS accounts can be used within the same collaboration
2. SQL Query and Data Processing
   - SQL queries are used to access and process the source data
   - The queries run within the AWS Clean Rooms collaboration boundaries
   - Processed data feeds into ML Input Channels for model training
3. ML Model Development
   - Source code for the model can be developed using AWS Deep Learning Container Images

- Custom container images must be created and stored in Amazon Elastic Container Registry

4. Infrastructure Components
   - Amazon Elastic Container Registry stores and manages the ML model containers
   - ML processing occurs within the secure AWS Clean Rooms collaboration environment
5. Monitoring and Logging
   - Amazon CloudWatch provides metrics and logs for both collaborating parties
   - Monitoring is available across AWS accounts involved in the collaboration
   - Performance metrics and operational logs are accessible to relevant parties
6. Results Management
   - Access to results is controlled according to collaboration permissions

Before you get started, see the Custom ML modeling prerequisites and Model authoring guidelines for the training container for more information.

Privacy-enhanced synthetic dataset generation

A *synthetic dataset* has similar statistical properties to the original dataset it's based on, but doesn't contain the real-world observations present in the original dataset. By using privacy-enhanced synthetic datasets, you can unlock new machine learning (ML) model training use cases that data privacy concerns previously prevented. When you create an ML input channel, you can generate synthetic data to protect sensitive information while training ML models.

When creating a template with synthetic data, you must:

- Require template output to be synthetic
- Classify output schema columns as numerical or categorical
- Customize synthetic data based on organizational needs
- Adjust privacy settings:
  - Set privacy level (epsilon)
  - Configure privacy threshold

Warning

Synthetic data generation protects against inferring individual attributes whether specific individuals are present in the original dataset or learning attributes of those individuals are present. However, it doesn't prevent literal values from the original dataset, including personally identifiable information (PII) from appearing in the synthetic dataset.
We recommend avoiding values in the input dataset that are associated with only one data subject because these may re-identify a data subject. For example, if only one user lives in a zip code, the presence of that zip code in the synthetic dataset would confirm that user was in the original dataset. Techniques like truncating high precision values or replacing uncommon catalogues with *other* can be used to mitigate this risk. These transformations can be part of the query used to create the ML input channel.

For more information about how to generate synthetic data for custom model training, see Creating a SQL analysis template.

Analysis templates with synthetic outputs can only be used to create ML input channels. For more information, see Creating an ML input channel in AWS Clean Rooms ML.

Considerations for synthetic data generation

With AWS Clean Rooms ML, collaboration members can create a synthetic dataset that irreversibly de-identifies the subjects of the original dataset from their collective datasets to train a custom machine learning model. When creating the collaboration, you must configure payment information to specify who pays for synthetic data generation. Here are the high-level steps to generate a synthetic dataset and train a custom machine learning model:

1. A collaboration member creates an analysis template that includes:
   - The SQL needed to define the dataset to be synthesized.
   - Privacy-related configurations used to ensure the synthetic data meets data providers' compliance requirements.
2. Once all the data providers approve the analysis template, the collaboration query runner creates an machine learning (ML) input channel, using the template.
3. Clean Rooms ML generates the synthetic dataset and verifies that it meets the privacy thresholds specified in the analysis template.
4. If all the thresholds are satisfied, the ML input channel is populated with the synthetic dataset.

5. Customers can then use this ML input channel to train the custom ML model associated to the collaboration.

Important considerations:

- Synthetic data generated in Clean Rooms ML does **not** remove, redact, obfuscate, or sanitize any individual values, including personally identifiable information (PII) found in the original dataset. The synthetic dataset is generated by sampling values, but not whole records, from the original dataset.
- If the original dataset contains similar rows, it's possible the synthetic data contains rows that look identical to rows in the original dataset.

Dataset preparation:

- Avoid columns with a significantly imbalanced class distribution. This is especially important for the predicted value or “Y” column. Extreme imbalances reduce the synthetic dataset's overall privacy.
- Clean Rooms ML doesn't support generating synthetic data from time series data where maintaining correlations across sequential records is important.
- Clean Rooms ML doesn't support generating synthetic data from text or unstructured data.
- The following data types are supported:

Data type name
BIGINT
BOOLEAN
CHAR
DATE
DECIMAL
FLOAT

Data type name
INTEGER
LONG
REAL
SHORT
SMALLINT
TIME
TIMESTAMP_LTZ
TIMESTAMP_NTZ
TINYINT
VARCHAR

Limitations:

- For synthetic data generation, the maximum number of predictive columns is one.
- If the target column is categorical, the maximum number of categories in the orginal dataset is 100.
- In the original dataset, the number of rows must be between 1,500 and 2.5 million and the maximum number of columns is 1.000. For non-null values in the target column, the minimum number of rows is 1,000.

Privacy metrics:

- Clean Rooms ML provides a privacy score that measures how protected the generated synthetic data is against membership inference attacks (MIAs). The service holds out 5% of the original data from the synthesization process to calculate this score.
- Scores near 50% are considered good; higher scores indicate less protection against MIAs. Scores significantly below 50% are rare and may be due to non-representation of patterns from the original data in the synthesized data.

Downstream custom model:

- Synthetic data generated in Clean Rooms ML is best suited for training binary classification models and multi-class classification models with up to five classes.
- Training regression models using synthetic data generated in Clean Rooms ML may result in low model accuracy, as measured by Root Mean Square Error (RMSE).

Creating and joining the collaboration in AWS Clean Rooms ML

The collaboration creator is responsible for creating the collaboration, inviting members, and assigning their roles. The invited members join the collaboration and specify results settings, trained model artifacts destination settings and accept payment responsibilities, depending on how the collaboration is set up.

Creating a collaboration for machine learning

The following procedure shows how to create a collaboration for machine learning, invite one or more members, and assign members that can start model training, receive results, receive trained model results, including model artifacts and metrics, and receive model inference results. The collaboration creator also assigns a member who will pay for query compute, model training, and model inference costs.

Console

To create a collaboration for machine learning (console)

1. Create a collaboration and invite one or more members to join the collaboration
2. Assign the following **Member abilities** for **Analysis using queries and jobs**:
    - Assign **Run queries** to the member who will start model training.

- Assign **Receive results from analysis** to the members who will receive query results.

3. Assign the following **Member abilities** for **ML modeling using purpose-built workflows**:

   - Assign **Receive output from trained models** to the member who will receive trained model results, including model artifacts and metrics.
   - Assign **Receive output from model inference** to the member who will receive the model inference results.

4. For **Configure payment**, specify the members who will pay for query compute, model training, model inference, and synthetic dataset generation costs.

   A synthetic dataset has similar statistical properties to the original dataset it's based on, but doesn't contain the real-world observations present in the original dataset. By using privacy-enhanced synthetic datasets you can unlock new ML model training use cases that data privacy concerns previously prevented. For more information, see Privacy-enhanced synthetic data generation.

   Each of these costs can be assigned to the same or different members. If an invited member is the member who is responsible to pay for payment costs, they must accept their payment responsibilities before joining the collaboration.

5. For **Configure membership**, the collaboration creator can decide to join the membership now or create a membership later. The collaboration creator must then set up the ML configuration.

   a. If the collaboration creator is also the results receiver, they must also specify the query results destination and format in the **Results settings defaults**.

   b. The ML configuration provides a role for Clean Rooms ML to publish metrics to an AWS account. If the collaboration creator is also receiving trained model artifacts, they can specify the Amazon S3 bucket used to receive results.

   c. In the **ML configurations** section, select **Create ML configuration** and then specify the **Model output destination on Amazon S3** and the **Service access role** needed to access this location.

   d. If the collaboration creator is the member who is responsible to pay for payment costs, they must accept their payment responsibilities before creating the collaboration.

API

To create a collaboration for machine learning (API)

1. Create a collaboration and invite one or more members to join the collaboration
2. Assign the following roles to collaboration members:
   - `CAN_QUERY` - assigned to the member who will start model training and inference.
   - `CAN_RECEIVE_MODEL_OUTPUT` - assigned to the members who will receive trained model results.
   - `CAN_RECEIVE_INFERENCE_OUTPUT` - assigned to the members who will receive model inference results.

   If the collaboration creator is also the results receiver, they must also specify the query results destination and format during collaboration creation. They also give a service role Amazon Resource Name (ARN) to write the results to the query results destination.
3. Specify the members who will pay for query compute, model training, and model inference costs. Each of these costs can be assigned to the same or different members. If an invited member is the member who is responsible to pay for payment costs, they must accept their payment responsibilities before joining the collaboration.
4. The following code creates a collaboration, invites a member that can run queries and receive results, and specifies the collaboration creator as the model artifacts receiver.

```
import boto3
acr_client= boto3.client('cleanrooms')

collaboration = a_acr_client.create_collaboration(
    members=[
        {
         'accountId': 'invited_member_accountId',
         'memberAbilities':["CAN_QUERY","CAN_RECEIVE_RESULTS"],
         'displayName': 'member_display_name'
        }
    ],
    name='collaboration_name',
    description=collaboration_description,
    creatorMLMemberAbilities= {
        'customMLMemberAbilities':["CAN_RECEIVE_MODEL_OUTPUT",
 "CAN_RECEIVE_INFERENCE_OUTPUT"],
    },
```

```
    creatorDisplayName='creator_display_name',
    queryLogStatus="ENABLED",
    analyticsEngine="SPARK",
    creatorPaymentConfiguration={
        "queryCompute": {
            "isResponsible": True
        },
        "machineLearning": {
            "modelTraining": {
                "isResponsible": True
            },
            "modelInference": {
                "isResponsible": True
            }
        }
    }
)

collaboration_id = collaboration['collaboration']['id']
print("collaborationId: {collaboration_id}")

member_membership = a_acr_client.create_membership(
    collaborationIdentifier = collaboration_id,
    queryLogStatus = 'ENABLED',
    paymentConfiguration={
        "queryCompute": {
            "isResponsible": True
        },
        "machineLearning": {
            "modelTraining": {
                "isResponsible": True
            },
            "modelInference": {
                "isResponsible": True
            }
        }
    }
)
```

5. The collaboration creator must then set up the ML configuration. The ML configuration provides a role for Clean Rooms ML to publish metrics and logs to an AWS account. If the collaboration creator is also receiving results (model artifacts or inference results), they can specify the Amazon S3 bucket used to receive results.

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.put_ml_configuration(
        membershipIdentifier=membership_id,
        defaultOutputLocation={
           'roleArn':'arn:aws:iam::account:role/roleName',
           'destination':{
                's3Destination':{
                    's3Uri':"s3://bucketName/prefix"
                }
            }
        }
    )
```

Joining a collaboration

After the collaboration creator has finished their tasks, the invited members must complete theirs.

Console

To create a membership and join a collaboration (console)

1. The invited member creates a membership and joins the collaboration.
2. If the invited member is the member who is responsible to pay, including query compute, model training, and model inference costs, they must accept their payment responsibilities before joining the collaboration.
3. The invited member sets up the ML configuration, which provides a role for Clean Rooms ML to publish model metrics to an AWS account. If they're also the member receiving trained model artifacts, they must give an Amazon S3 bucket where trained model artifacts are stored.

API

To create a membership and join a collaboration (API)

1. If the invited member is the member who can receive results, they specify the query results destination and format. They also provide a service role ARN that allows the service to write to the query results destination

   If the invited member is the member who is responsible to pay, including query compute, model training, and model inference costs, they must accept their payment responsibilities before joining the collaboration.

   If the invited member is the member who is responsible to pay for model training and model inference for custom modeling, they must accept their payment responsibilities before joining the collaboration.

   The following code creates a membership with query logging enabled.

```
import boto3
acr_client= boto3.client('cleanrooms')

acr_client.create_membership(
    membershipIdentifier='membership_id',
    queryLogStatus='ENABLED'
)
```

2. The invited member sets up the ML configuration, which provides a role for Clean Rooms ML to publish model metrics to an AWS account. If they're also the member receiving trained model artifacts, they must provide an Amazon S3 bucket where trained model artifacts are stored.

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.put_ml_configuration(
        membershipIdentifier='membership_id',
        defaultOutputLocation={
           'roleArn':"arn:aws:iam::account:role/role_name",
           'destination':{
                's3Destination':{
                    's3Uri':"s3://bucket_name/prefix"
                }
            }
        }
    )
```

Contributing training data in AWS Clean Rooms ML

After the collaboration creator has created the collaboration and invited members have joined, you are ready to contribute training data to the collaboration. Any member can contribute training data.

Console

To contribute training data (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Tables**.
3. On the **Tables** page, choose **Configure new table**.
4. For **Configure new table**, for **Data source,** choose **Amazon S3**, **Amazon Athena**, or **Snowflake** and complete the following steps, based on your data source:

If you're using	Then
Amazon S3	1. Choose a **Database** from the dropdown list, and then select the **Table** from the database. 2. For **Columns allowed in collaborations**, choose either **All columns** or **Custom list**. 3. For **Configured table details**, provide the **Name** and an optional **Description** for this table. 4. If you want to report model metrics, enter the **Name** of the metrics and the **Regex** statement that will search the output logs to find the metric. 5. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

If you're using	Then
Amazon Athena	1. Choose a **Database** from the dropdown list, and then select the **Table** from the database. 2. For **Columns allowed in collaborations**, choose either **All columns** or **Custom list**. 3. For **Configured table details**, provide the **Name** and an optional **Description** for this table. 4. If you want to report model metrics, enter the **Name** of the metrics and the **Regex** statement that will search the output logs to find the metric. 5. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
Snowflake	1. Specify the **Snowflake credentials** using an existing secret ARN or storing a new secret for this table. 2. For **Snowflake table and schema details**, enter the details manually or automatically import the details. 3. For the **Schema**, enter the **Column name** and choose the **Data type** from the dropdown list. 4. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.

5. Choose **Configure new table**.

6. On the table details page, choose **Configure analysis rule** to configure a custom analysis rule for this table. A custom analysis rule limits access to your data. You can either allow a specific set of pre-authorized queries on your data or allow a specific set of accounts to query your data.

   a. For **Analysis rule type**, choose **Custom** and for **Creation method**, choose **Guided flow**.

   b. Choose **Next**.

   c. For **Specify analysis controls**, choose between **Review each new analysis** and **Allow any analysis by specific collaborators**.

   d. Choose **Next**.

   e. (Optional) For **Specify analysis results controls**, for **Columns not allowed in output** specify whether you want to exclude any columns from the output. If you choose **None**, no columns are excluded from the output. If you choose **Custom list**, you can specify certain columns that will be removed from the output.

   f. For **Additional analyses applied to output** specify whether you want to allow, deny, or require an additional analysis before results are generated.

   g. Choose **Next**.

   h. (Optional) For **Set differential privacy**, choose **Turn off**.

   i. Choose **Next**.

   j. Review the information on the **Review and configure** page, then choose **Configure analysis rule**.

7. From the table details page, choose **Associate to collaboration**.

8. In the **Associate table** dialog box, select the collaboration that you want to associate this table to and choose **Choose collaboration**.

9. On the **Associate table** page, review and verify the information in **Table association details**, **Service access**, and **Tags**. Choose **Associate table**.

10. In the **Tables associated by you** table, select the radio button next to the table that you just associated. From the **Actions** menu, choose **Configure** in the **Collaboration analysis rule** group.

11. On the **Configure collaboration analysis rule** page, for **Allowed additional analyses**, choose whether any collaboration members or specific collaboration members can perform additional analyses.

For **Results delivery**, choose which members are allowed to receive results from query outputs.

12. Choose **Configure analysis rule**.

API

To contribute training data (API)

1. Configure an existing AWS Glue table for use in AWS Clean Rooms by providing the table and the columns that can be used.

   Run the following code with your specific parameters.

```
import boto3
acr_client= boto3.client('cleanrooms')

acr_client.create_configured_table(
    name='configured_table_name',
    tableReference= {
        'glue': {
            'tableName': 'glue_table_name',
            'databaseName': 'glue_database_name'
        }
    },
    analysisMethod="DIRECT_QUERY",
    allowedColumns=["column1", "column2", "column3",...]
)
```

2. Configure a custom analysis rule that limits access to your data. You can either allow a specific set of pre-authorized queries on your data or allow a specific set of accounts to query your data.

   Run the following code with your specific parameters.

```
import boto3
acr_client= boto3.client('cleanrooms')

acr_client.create_configured_table_analysis_rule(
    configuredTableIdentifier='configured_table_id',
    analysisRuleType='CUSTOM',
    analysisRulePolicy= {
```

```
        'v1': {
            'custom': {
                'allowedAnalyses': ['ANY_QUERY'],
                'allowedAnalysisProviders': ['query_runner_account'],
                'additionalAnalyses': "REQUIRED"
            }
        }
    }
)
```

In this example, a specific account is allowed to run any query on the data and an additional analysis is required.

3. Associate a configured table to the collaboration and provide a service access role to the AWS Glue tables.

   Run the following code with your specific parameters.

```
import boto3
acr_client= boto3.client('cleanrooms')

acr_client.create_configured_table_association(
    name='configured_table_association_name',
    membershipIdentifier='membership_id',
    configuredTableIdentifier='configured_table_id',
    roleArn='arn:aws:iam::account:role/role_name'
)
```

> **Note**
>
> This service role has permissions to the tables. The service role is assumable only by AWS Clean Rooms to run allowed queries on behalf of the member who can query. No collaboration members (other than the data owner) have access to the underlying tables in the collaboration. The data owner can turn off differential privacy to make their tables available for querying by other members.

4. Finally, add an analysis rule to the configured table association.

   Run the following code with your specific parameters.

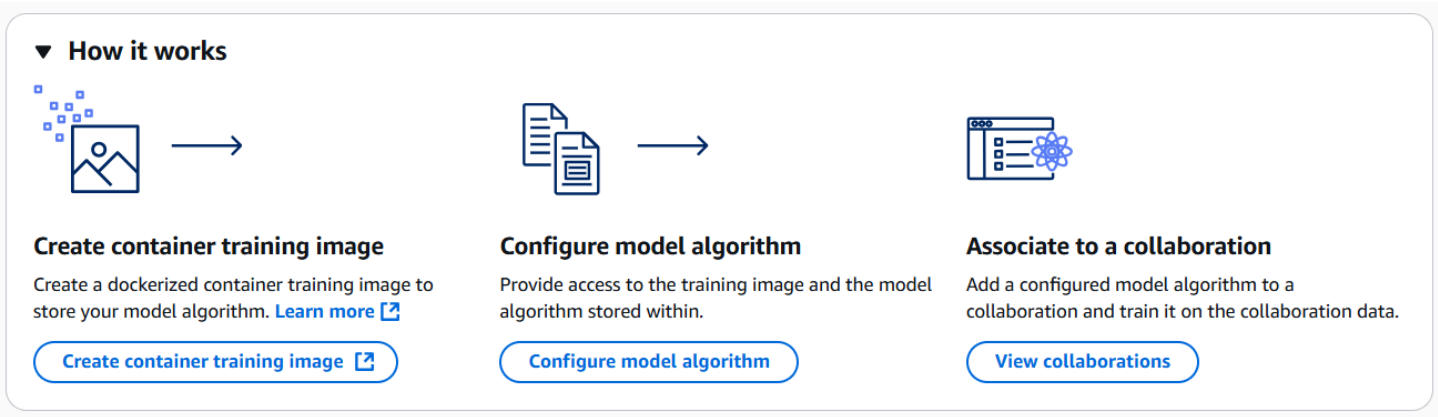
```
import boto3
```

```
acr_client= boto3.client('cleanrooms')

acr_client.create_configured_table_association_analysis_rule(

 configuredTableAssociationIdentifier='configured_table_association_identifier',
    membershipIdentifier='membership_id',
    configuredTableIdentifier='configured_table_id',
    analysisRuleType = 'CUSTOM',
    analysisRulePolicy= {
        'v1': {
            'custom': {
                'allowedAdditionalAnalyses':
 ['configured_model_algorithm_association_arns'],
                'allowedResultReceivers': ['query_runner_account']
            }
        }
    }
)
```

Configuring a model algorithm in AWS Clean Rooms ML

After you have created a container training image, you must configure your model algorithm. Configuring a model algorithm makes it available for association to a collaboration.

The following image shows configuring a model algorithm as a step that happens after you create the container training image and before you associate it with the collaboration.



Console

To configure a custom ML model algorithm (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Custom ML models**.
3. On the **Custom ML models** page, choose **Configure model algorithm**.
4. On the **Configure model algorithm** page, for **Model algorithm details**, enter a **Name** and optional **Description**.
5. If you want to perform model training, for **Training image ECR container details**,
   a. Select the **Specify training image URI** checkbox.
   b. Select the **Repository** that has the training model, inference container, or both, from the dropdown list.
   c. Select the **Image**.
   d. (Optional) Enter the **Value** for the **Entrypoints** to access the training image.
   e. (Optional) Enter the **Value** for the **Arguments**.
6. (Optional) If you want to report model metrics, for **Training metrics**, enter the **Name** of the metrics and the **Regex** statement that will search the output logs to find the metric.
7. If you want to perform model inference, for **Inference image ECR container details**,
   a. Select the **Specify inference image URI** checkbox.
   b. Select the **Repository** from the dropdown list.
   c. Select the **Image**.
8. For **Service access**, choose the **Existing service role name** that will be used to access this table.
9. For **Encryption**, choose the **Customize encryption settings** to specify your own KMS key and related information. Otherwise, Clean Rooms ML will manage the encryption.
10. If you want to enable **Tags**, choose **Add new tag** and then enter the **Key** and **Value** pair.
11. Choose **Configure model algorithm**.

API

To configure a custom ML model algorithm (API)

1. Create a SageMaker AI compatible docker image. Clean Rooms ML only supports SageMaker AI compatible docker images.
2. After you have created a SageMaker AI compatible docker image, use Amazon ECR to create a training image. Follow the directions in Amazon Elastic Container Registry User Guide to create a container training image.
3. Configure the model algorithm for use in Clean Rooms ML. You must give the following information:
   - The Amazon ECR repository link and additional arguments to train the model and run inference. Clean Rooms ML supports running batch transform jobs on an inference container.
   - A service access role that allows Clean Rooms ML to access the repository.
   - (Optional) An inference container. Although you can provide this in a separate configured model algorithm, we recommend that you provide it in this step so that both the training and inference container are managed as part of the same resource.

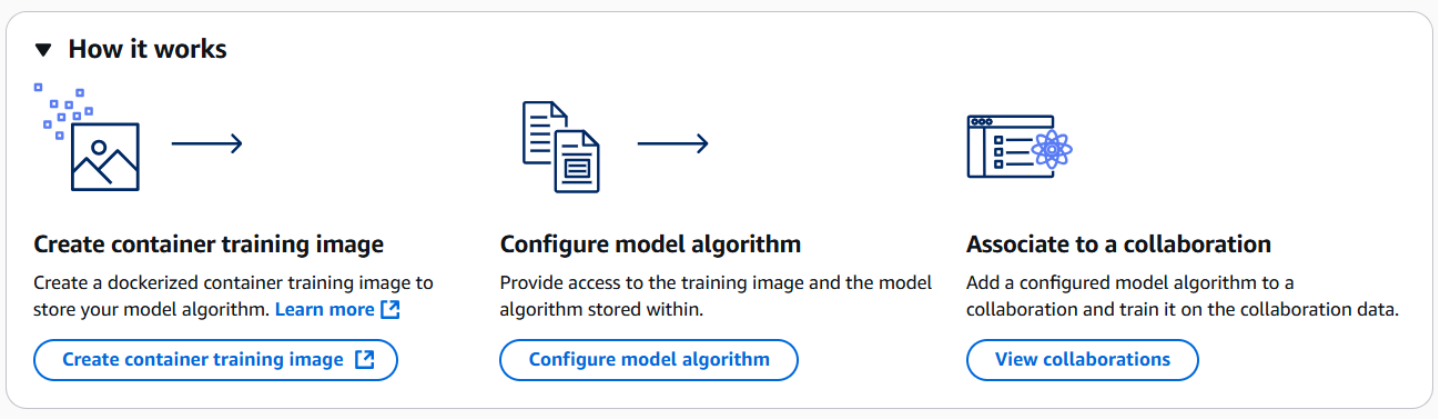
   Run the following code with your specific parameters.

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.create_configured_model_algorithm(
    name='configured_model_algorithm_name',
    trainingContainerConfig={
        'imageUri': 'account.dkr.ecr.region.amazonaws.com/image_name:tag',
        'metricDefinitions': [
            {
                'name': 'custom_metric_name_1',
                'regex': 'custom_metric_regex_1'
            }
        ]
    },
    inferenceContainerConfig={
        'imageUri':'account.dkr.ecr.region.amazonaws.com/image_name:tag',
    }
    roleArn='arn:aws:iam::account:role/role_name'
)
```

Associating the configured model algorithm in AWS Clean Rooms ML

After you have configured the model algorithm, you are ready to associate the model algorithm to a collaboration. Associating a model algorithm makes the model algorithm available to all members of the collaboration.

The following image shows associating the configured model algorithm as the last step, after creating the container training image and configuring a model algorithm.



Console

> **Note**
>
> After the model algorithm is associated, it can't be edited. To make changes, you can delete the associated model algorithm and associate a new one.

To associate a custom ML model algorithm (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Custom ML models**.
3. On the **Custom ML models** page, choose the configured model algorithm that you want to associate to a collaboration and then choose **Associate to collaboration**.
4. In the **Associate configured model algorithm** window, choose the **Collaboration** that you want to associate to.

5. Choose **Choose collaboration**.
6. On the **Associate model algorithm** page, for **Model algorithm association details**, enter a **Name** and optional **Description**.
7. For **Model algorithm**, choose a **Configured model algorithm**.
8. For **Trained model export privacy configurations**,
   a. To export model files, select the **Model files** checkbox.
   b. To export output files, select the **Output files** checkbox.
   c. Enter a **Max size value** for the exported data. The value must be between 0.01 and 10.
9. (Optional) If you want to send either full error logs or shorter error summaries to members, under **Trained model inference job privacy configuration**,
   a. Under **Full logs**, select one or more **Account IDs** from the dropdown list.
   b. (Optional) If you want to send logs that match a filter pattern, enter a **Filter pattern**.
   c. (Optional) If you want to add another account and optional filter pattern, choose **Add log policy**.
   d. Under **Error summaries**, select one or more **Account IDs** from the dropdown list.
   e. (Optional) Select one or more **Entities to redact** to specify which entities will be redacted from the error log or error summaries.
      - **PII** – redact Personally Identifiable Information
      - **Numbers** – redact numbers
      - **Custom** – redact based on the custom redaction pattern

      i. If you chose **Custom** in the previous step, enter a **Custom redaction pattern**. This logs information that matches this pattern.
      ii. (Optional) If you want to add another custom redaction pattern, choose **Add another custom pattern**.
10. (Optional) If you want to configure trained model metrics, under **Trained model metrics configuration**, select a **Noise level** from the dropdown list.

    You can choose **None**, **Low**, **Medium** and **High**.
11. (Optional) If you want to set the maximum artifacts size, under **Artifacts configuration**, enter the **Max artifacts size value**. The value must be between 0.01 and 10.

12. (Optional) If you want to enable **Tags**, choose **Add new tag** and then enter the **Key** and **Value** pair.
13. Choose **Associate**.

API

To associate a custom ML model algorithm (API)

Run the following code with your specific parameters.

You also provide a privacy policy that defines who has access to the different logs, allows customers to define regex, and how much data can be exported from the training model outputs or inference results.

> **Note**
>
> Configured model algorithm associations are immutable.

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.create_configured_model_algorithm_association(
    name='configured_model_algorithm_association_name',
    description='purpose of the association',
    configuredModelAlgorithmArn='arn:aws:cleanrooms-ml:region:account:membership/
membershipIdentifier/configured-model-algorithm/identifier',
    privacyConfiguration={
        "policies": {
            "trainedModelExports": {
                "filesToExport": ['files to export'],
                "containerLogs": [
                    {
                        "allowedAccountIds": ['member_account_id'],
                        "filterPattern": ['filter pattern'],
                        "logRedactionConfiguration": {
                            "entitiesToRedact": [
                                'ALL_PERSONALLY_IDENTIFIABLE_INFORMATION',
                                'NUMBERS',
                                'CUSTOM'
                            ],
```

```
                            "customEntityConfig": {
                                "customDataIdentifiers": [
                                    'custom_regex_1',
                                    'custom_regex_2'
                                ]
                            }
                        }
                    }
                ],
                "containerMetrics": {
                    "noiseLevel": 'noise value'
                },
                "maxArtifactSize": {
                    "unit": 'unit',
                    "value": 'number'
                }
            },
            "trainedModelInferenceJobs": {
                "containerLogs": [
                    {
                        "allowedAccountIds": ['member_account_id'],
                        "filterPattern": ['filter pattern'],
                        "logRedactionConfiguration": {
                            "entitiesToRedact": [
                                'ALL_PERSONALLY_IDENTIFIABLE_INFORMATION',
                                'NUMBERS',
                                'CUSTOM'
                            ],
                            "customEntityConfig": {
                                "customDataIdentifiers": [
                                    'custom_regex_1',
                                    'custom_regex_2'
                                ]
                            }
                        }
                    }
                ],
                "maxOutputSize": {
                    "unit": 'unit',
                    "value": 'number'
                }
            }
        }
    },
```

```
    tags={
        'tag': 'tag'
    }
)
```

After the configured model algorithm is associated to the collaboration, training data providers must add a collaboration analysis rule to their table. This rule allows the configured model algorithm association to access their configured table. All contributing training data providers must run the following code:

```
import boto3
acr_client= boto3.client('cleanrooms')

acr_client.create_configured_table_association_analysis_rule(
    membershipIdentifier= 'membership_id',
    configuredTableAssociationIdentifier= 'configured_table_association_id',
    analysisRuleType= 'CUSTOM',
    analysisRulePolicy = {
        'v1': {
            'custom': {
                'allowedAdditionalAnalyses': ['arn:aws:cleanrooms-
ml:region:*:membership/*/configured-model-algorithm-association/*''],
                'allowedResultReceivers': []
            }
        }
    }
)
```

> **Note**
>
> Because configured model algorithm associations are immutable, we recommend that training data providers who wants to allowlist models for use to use wild cards in `allowedAdditionalAnalyses` during the first few iterations of custom model configuration. This allows model providers to iterate on their code without requiring other training providers to re-associate before training their updated model code with data.

Creating an ML input channel in AWS Clean Rooms ML

Prerequisites:

- An AWS account with access to AWS Clean Rooms
- A collaboration set up in AWS Clean Rooms where you want to create the ML input channel
- Permissions to query data and create ML input channels in the collaboration.
- (Optional) An existing model algorithm to associate with the ML input channel, or permissions to create a new one
- (Optional) Tables with analysis rules that can be run for your specified model.
- (Optional) An existing SQL query or analysis template to use for generating the dataset
- (Optional) An existing service role with appropriate permissions, or permissions to create a new service role
- (Optional) A custom AWS KMS key if you want to use your own encryption key
- Appropriate permissions to create and manage ML models in the collaboration

An *ML input channel* is a dataset that is created from a specific data query. Members with the ability to query data can prepare their data for training and inference by creating an ML input channel. Creating an ML input channel allows that data to be used in different training models within the same collaboration. You should create separate ML input channels for training and inference.

To create an ML input channel, you must specify the SQL query that is used to query the input data and create the ML input channel. The results of this query are never shared with any member and remain within the boundaries of Clean Rooms ML. The reference Amazon Resource Name (ARN) is used in the next steps to train a model or run inference.

Console

To create an ML input channel (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration where you want to create an ML input channel.

4. After the collaboration opens, choose the **ML models** tab.
5. Under **Custom ML models**, in the **ML input channels** section, choose **Create ML input channel**.
6. On the **Create ML input channel** page, for **ML input channel details**, do the following:
   a. For **Name**, enter a unique name for your channel.
   b. (Optional) For **Description**, enter a description of your channel.
   c. For **Associated model algorithm**, select the algorithm to use.

      Choose **Associate model algorithm** to add a new one.
7. For **Dataset**, choose a method to generate the training dataset:
   - Choose **SQL query** to use the results of a SQL query as the training dataset.

     If you chose **SQL query**, enter your query in the **SQL query** field.

     (Optional) To import a query you've used recently, choose **Import from recent queries**.
   - Choose **Analysis template** to use the results of an analysis template as the training dataset.

> **Warning**
>
> Synthetic data generation protects against inferring individual attributes whether specific individuals are present in the original dataset or learning attributes of those individuals are present. However, it doesn't prevent literal values from the original dataset, including personally identifiable information (PII) from appearing in the synthetic dataset.
> We recommend avoiding values in the input dataset that are associated with only one data subject because these may re-identify a data subject. For example, if only one user lives in a zip code, the presence of that zip code in the synthetic dataset would confirm that user was in the original dataset. Techniques like truncating high precision values or replacing uncommon catalogues with *other* can be used to mitigate this risk. These transformations can be part of the query used to create the ML input channel.

a. If no tables are associated, choose **Associate table** to add tables with an analysis rule that can be run for the specified model.

b. Choose **Worker type** to use when creating this data channel. The default worker type is **CR.1X**. Specify the **Number of workers** to use. The default worker number is **16**. To specify **Spark properties**:

   i. Expand **Spark properties**.

   ii. Choose **Add Spark properties**.

   iii. On the **Spark properties** dialog box, choose a **Property name** from the dropdown list and enter a **Value**.

The following tables provide a definition for each property.

For more information about Spark properties, see Spark Properties in the Apache Spark documentation.

> **Note**
>
> You can configure a maximum of 50 Spark properties. Each property value can be up to 500 characters.

Property Name	Description	Default Value
spark.task.maxFailures	Controls how many consecutive times a task can fail before the job fails. Requires a value greater than or equal to 1. The number of allowed retries equals this value minus 1. The failure count resets if any attempt succeeds. Failures across different tasks don't accumulate toward this limit.	4

Property Name	Description	Default Value
spark.sql.files.maxPartitionBytes	Sets the maximum number of bytes to pack into a single partition when reading from file-based sources such as Parquet, JSON, and ORC.	128MB
spark.hadoop.fs.s3.maxRetries	Sets the maximum number of retry attempts for Amazon S3 file operations.	(none)
spark.network.timeout	Sets the default timeout for all network interactions. Overrides the following timeout settings if they aren't configured: • spark.storage.blockManagerHeartbeatTimeoutMs • spark.shuffle.io.connectionTimeout • spark.rpc.askTimeout • spark.rpc.lookupTimeout	120s
spark.rdd.compress	Specifies whether to compress serialized RDD partitions using spark.io.compression.codec. Applies to StorageLevel.MEMORY_ONLY_SER in Java and Scala, or StorageLevel.MEMORY_ONLY in Python. Reduces storage space but requires additional CPU processing time.	false
spark.shuffle.spill.compress	Specifies whether to compress shuffle spill data using spark.io.compression.codec.	true
spark.shuffle.compress	Specifies whether to compress map output files. Compression uses spark.io.compression.codec.	true

Property Name	Description	Default Value
spark.shuffle.service.index.cache.size	Sets the cache size limit, in bytes unless otherwise specified.	100m
spark.shuffle.io.maxRetries	Sets the maximum number of retries for fetches that fail due to IO-related exceptions.	3
spark.shuffle.io.retryWait	Sets the wait time between retries of fetches. The maximum delay caused by retrying is 15 seconds by default, calculated as maxRetries * retryWait.	5s
spark.shuffle.io.connectionTimeout	Sets the timeout for established connections between shuffle servers and clients to be marked as idle and closed if there are still outstanding fetch requests but no traffic on the channel.	(value of spark.network.timeout)
spark.driver.maxResultSize	Sets the total size limit of serialized results of all partitions for each Spark action, in bytes. Should be at least 1M, or 0 for unlimited.	1g
spark.memory.fraction	Sets the fraction of (heap space - 300MB) used for execution and storage. The lower this value, the more frequently spills and cached data eviction occur. Leaving this at the default value is recommended.	0.6
spark.scheduler.mode	Sets the scheduling mode between jobs submitted to the same SparkContext. Can be set to FAIR to use fair sharing instead of queueing jobs one after another. Supported values: FAIR, FIFO.	FIFO

Property Name	Description	Default Value
spark.sql.adaptive.advisoryPartitionSizeInBytes	Sets the target size in bytes for shuffle partitions during adaptive optimization when spark.sql.adaptive.enabled is true. Controls partition size when coalescing small partitions or splitting skewed partitions.	(value of spark.sql.adaptive.shuffle.targetPostShuffleInputSize)
spark.sql.adaptive.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Applies only in adaptive framework. Uses the same default value as spark.sql.autoBroadcastJoinThreshold. Set to -1 to disable broadcasting.	(none)
spark.sql.adaptive.coalescePartitions.enabled	Specifies whether to coalesce contiguous shuffle partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes to optimize task size. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Defines the initial number of shuffle partitions before coalescing. Requires both spark.sql.adaptive.enabled and spark.sql.adaptive.coalescePartitions.enabled to be true. Defaults to the value of spark.sql.shuffle.partitions.	(none)
spark.sql.adaptive.coalescePartitions.minPartitionSize	Sets the minimum size for coalesced shuffle partitions to prevent partitions from becoming too small during adaptive optimization.	1 MB

Property Name	Description	Default Value
spark.sql.adaptive.coalescePartitions.parallelismFirst	Specifies whether to calculate partition sizes based on cluster parallelism instead of spark.sql.adaptive.advisoryPartitionSizeInBytes during partition coalescing. Generates smaller partition sizes than the configured target size to maximize parallelism. We recommend setting this to false on busy clusters to improve resource utilization by preventing excessive small tasks.	true
spark.sql.adaptive.enabled	Specifies whether to enable adaptive query execution to re-optimize query plans during query execution, based on accurate runtime statistics.	true
spark.sql.adaptive.forceOptimizeSkewedJoin	Specifies whether to force enable OptimizeSkewedJoin even if it introduces extra shuffle.	false
spark.sql.adaptive.localShuffleReader.enabled	Specifies whether to use local shuffle readers when shuffle partitioning isn't required, such as after converting from sort-merge joins to broadcast-hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Sets the maximum partition size in bytes for building local hash maps. Prioritizes shuffled hash joins over sort-merge joins when: • This value equals or exceeds spark.sql.adaptive.advisoryPartitionSizeInBytes • All partition sizes are within this limit Overrides spark.sql.join.preferSortMergeJoin setting.	0 bytes
spark.sql.adaptive.optimizeSkewsInRebalancePartitions.enabled	Specifies whether to optimize skewed shuffle partitions by splitting them into smaller partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.rebalancePartitionsSmallPartitionFactor	Defines the size threshold factor for merging partitions during splitting. Partitions smaller than this factor multiplied by spark.sql.adaptive.advisoryPartitionSizeInBytes are merged.	0.2
spark.sql.adaptive.skewJoin.enabled	Specifies whether to handle data skew in shuffled joins by splitting and optionally replicating skewed partitions. Applies to sort-merge and shuffled hash joins. Requires spark.sql.adaptive.enabled to be true.	true

Property Name	Description	Default Value
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Determines the size factor that determines partition skew. A partition is skewed when its size exceeds both: • This factor multiplied by the median partition size • The value of spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Sets the size threshold in bytes for identifying skewed partitions. A partition is skewed when its size exceeds both: • This threshold • The median partition size multiplied by spark.sql.adaptive.skewJoin.skewedPartitionFactor We recommend setting this value larger than spark.sql.adaptive.advisoryPartitionSizeInBytes.	256MB
spark.sql.broadcastTimeout	Controls the timeout period in seconds for the broadcast operations during broadcast joins.	300 seconds
spark.sql.cbo.enabled	Specifies whether to enable cost-based optimization (CBO) for plan statistics estimation.	false
spark.sql.cbo.joinReorder.dp.star.filter	Specifies whether to apply star-join filter heuristics during cost-based join enumeration.	false

Property Name	Description	Default Value
spark.sql.cbo.joinReorder.dp.threshold	Sets the maximum number of joined nodes allowed in the dynamic programming algorithm.	12
spark.sql.cbo.joinReorder.enabled	Specifies whether to enable join reordering in cost-based optimization (CBO).	false
spark.sql.cbo.planStats.enabled	Specifies whether to fetch row counts and column statistics from the catalog during logical plan generation.	false
spark.sql.cbo.starSchemaDetection	Specifies whether to enable join reordering based on star schema detection.	false
spark.sql.files.maxPartitionNum	Sets the target maximum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Rescales partitions when the initial count exceeds this value. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.files.maxRecordsPerFile	Sets the maximum number of records to write to a single file. No limit applies when set to zero or a negative value.	0
spark.sql.files.minPartitionNum	Sets the target minimum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Defaults to spark.sql.leafNodeDefaultParallelism. This is a suggested target, not a guaranteed limit.	(none)

Property Name	Description	Default Value
spark.sql.inMemoryColumnarStorage.batchSize	Controls the batch size for columnar caching. Increasing the size improves memory utilization and compression but increases the risk of out-of-memory errors.	10000
spark.sql.inMemoryColumnarStorage.compressed	Specifies whether to automatically select compression codecs for columns based on data statistics.	true
spark.sql.inMemoryColumnarStorage.enableVectorizedReader	Specifies whether to enable vectorized reading for columnar caching.	true
spark.sql.legacy.allowHashOnMapType	Specifies whether to allow hash operations on map type data structures. This legacy setting maintains compatibility with older Spark versions' map type handling.	(none)
spark.sql.legacy.allowNegativeScaleOfDecimal	Specifies whether to allow negative scale values in decimal type definitions. This legacy setting maintains compatibility with older Spark versions that supported negative decimal scales.	(none)
spark.sql.legacy.castComplexTypesToString.enabled	Specifies whether to enable legacy behavior for casting complex types to strings. Maintains compatibility with older Spark versions' type conversion rules.	(none)

Property Name	Description	Default Value
spark.sql.legacy.charVarcharAsString	Specifies whether to treat CHAR and VARCHAR types as STRING types. This legacy setting provides compatibility with older Spark versions' string type handling.	(none)
spark.sql.legacy.createEmptyCollectionUsingStringType	Specifies whether to create empty collections using string type elements. This legacy setting maintains compatibility with older Spark versions' collection initialization behavior.	(none)
spark.sql.legacy.exponentLiteralAsDecimal.enabled	Specifies whether to interpret exponential literals as decimal types. This legacy setting maintains compatibility with older Spark versions' numeric literal handling.	(none)
spark.sql.legacy.json.allowEmptyString.enabled	Specifies whether to allow empty strings in JSON processing. This legacy setting maintains compatibility with older Spark versions' JSON parsing behavior.	(none)
spark.sql.legacy.parquet.int96RebaseModeInRead	Specifies whether to use legacy INT96 timestamp rebase mode when reading Parquet files. This legacy setting maintains compatibility with older Spark versions' timestamp handling.	(none)
spark.sql.legacy.timeParserPolicy	Controls the time parsing behavior for backwards compatibility. This legacy setting determines how timestamps and dates are parsed from strings.	(none)

Property Name	Description	Default Value
spark.sql.legacy.typeCoercion.datetimeToString.enabled	Specifies whether to enable legacy type coercion behavior when converting datetime values to strings. Maintains compatibility with older Spark versions' datetime conversion rules.	(none)
spark.sql.maxSinglePartitionBytes	Sets the maximum partition size in bytes. The planner introduces shuffle operations for larger partitions to improve parallelism.	128m
spark.sql.metadataCacheTTLSeconds	Controls the time-to-live (TTL) for metadata caches. Applies to partition file metadata and session catalog caches. Requires: • A positive value greater than zero • spark.sql.catalogImplementation set to hive • spark.sql.hive.filesourcePartitionFileCacheSize greater than zero • spark.sql.hive.manageFilesourcePartitions set to true	-1000ms
spark.sql.optimizer.collapseProjectAlwaysInline	Specifies whether to collapse adjacent projections and inline expressions, even when it causes duplication.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	Specifies whether to generate predicates for partition columns used as join keys.	true

Property Name	Description	Default Value
spark.sql.optimizer.enableCsvExpressionOptimization	Specifies whether to optimize CSV expressions in SQL optimizer by pruning unnecessary columns from from_csv operations.	true
spark.sql.optimizer.enableJsonExpressionOptimization	Specifies whether to optimize JSON expressions in SQL optimizer by: • Pruning unnecessary columns from from_json operations • Simplifying from_json and to_json combinations • Optimizing named_struct operations	true
spark.sql.optimizer.excludedRules	Defines optimizer rules to disable, identified by comma-separated rule names. Some rules cannot be disabled as they are required for correctness. The optimizer logs which rules are successfully disabled.	(none)
spark.sql.optimizer.runtime.bloomFilter.applicationSideScanSizeThreshold	Sets the minimum aggregated scan size in bytes required to inject a Bloom filter on the application side.	10GB
spark.sql.optimizer.runtime.bloomFilter.creationSideThreshold	Defines the maximum size threshold for injecting a Bloom filter on the creation side.	10MB
spark.sql.optimizer.runtime.bloomFilter.enabled	Specifies whether to insert a Bloom filter to reduce shuffle data when one side of a shuffle join has a selective predicate.	true

Property Name	Description	Default Value
spark.sql.optimizer.runtime.bloomFilter.expectedNumItems	Defines the default number of expected items in the runtime Bloom filter.	1000000
spark.sql.optimizer.runtime.bloomFilter.maxNumBits	Sets the maximum number of bits allowed in the runtime Bloom filter.	67108864
spark.sql.optimizer.runtime.bloomFilter.maxNumItems	Sets the maximum number of expected items allowed in the runtime Bloom filter.	4000000
spark.sql.optimizer.runtime.bloomFilter.numBits	Defines the default number of bits used in the runtime Bloom filter.	8388608
spark.sql.optimizer.runtime.rowLevelOperationGroupFilter.enabled	Specifies whether to enable runtime group filtering for row-level operations. Allows data sources to: • Prune entire groups of data (such as files or partitions) using data source filters • Execute runtime queries to identify matching records • Discard unnecessary groups to avoid expensive rewrites Limitations: • Not all expressions can convert to data source filters • Some expressions require Spark evaluation (such as subqueries)	true

Property Name	Description	Default Value
spark.sql.optimizer.runtimeFilter.number.threshold	Sets the total number of injected runtime filters (non-DPP). This is to prevent driver OOMs with too many Bloom filters.	10
spark.sql.optimizer.runtimeFilter.semiJoinReduction.enabled	Specifies whether to insert a semi-join to reduce shuffle data when one side of a shuffle join has a selective predicate.	false
spark.sql.parquet.aggregatePushdown	Specifies whether to push down aggregates to Parquet for optimization. Supports: • MIN and MAX for boolean, integer, float, and date types • COUNT for all data types Throws an exception if statistics are missing from any Parquet file footer.	false
spark.sql.parquet.columnarReaderBatchSize	Controls the number of rows in each Parquet vectorized reader batch. Choose a value that balances performance overhead and memory usage to prevent out-of-memory errors.	4096
spark.sql.parquet.enableVectorizedReader	Specifies whether to enable vectorized Parquet decoding.	true

Property Name	Description	Default Value
spark.sql.shuffle.partitions	Sets the default number of partitions for data shuffling during joins or aggregations. Cannot be modified between structured streaming query restarts from the same checkpoint location.	200
spark.sql.shuffledHashJoinFactor	Defines the multiplication factor used to determine shuffle hash join eligibility. A shuffle hash join is selected when the small-side data size multiplied by this factor is less than the large-side data size.	3
spark.sql.sources.parallelPartitionDiscovery.threshold	Sets the maximum number of paths for driver-side file listing with file-based sources (Parquet, JSON, and ORC). When exceeded during partition discovery, files are listed using a separate Spark distributed job.	32
spark.sql.statistics.histogram.enabled	Specifies whether to generate equi-height histograms during column statistics computation to improve estimation accuracy. Requires an additional table scan beyond the one needed for basic column statistics.	false
spark.dynamicAllocation.executorIdleTimeout	Sets the duration an executor must be idle before it is removed when dynamic allocation is enabled.	60s
spark.dynamicAllocation.schedulerBacklogTimeout	Sets the duration that pending tasks must be backlogged before new executors are requested when dynamic allocation is enabled.	1s

Property Name	Description	Default Value
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Same as spark.dynamicAllocation.schedulerBacklogTimeout, but used only for subsequent executor requests.	(value of spark.dynamicAllocation.schedulerBacklogTimeout)
spark.scheduler.minRegisteredResourcesRatio	Sets the minimum ratio of registered resources (registered resources / total expected resources) to wait for before scheduling begins. Specified as a double between 0.0 and 1.0. Regardless of whether the minimum ratio of resources has been reached, the maximum amount of time it will wait before scheduling begins is controlled by spark.scheduler.maxRegisteredResourcesWaitingTime.	0.8
spark.scheduler.maxRegisteredResourcesWaitingTime	Sets the maximum amount of time to wait for resources to register before scheduling begins.	30s
spark.sql.hive.metastorePartitionPruningFallbackOnException	Specifies whether to fall back to getting all partitions from Hive metastore and perform partition pruning on the Spark client side when encountering MetaException from the metastore.	false
spark.sql.crossJoin.enabled	Specifies whether to allow queries that contain a cartesian product without explicit CROSS JOIN syntax.	true

Property Name	Description	Default Value
spark.sql.analyzer.maxIterations	Sets the maximum number of iterations the query analyzer runs before giving up. Higher values allow the analyzer to process very large or deeply nested queries.	100
spark.sql.datapreffetch.filescan.maxParallelismPerTask	Sets the maximum number of file splits to pre-fetch concurrently for each task when scanning files.	4
spark.sql.iceberg.data-prefetch.enabled	Specifies whether to enable data pre-fetch optimization when reading Iceberg tables.	true
spark.sql.legacy.nullValueWrittenAsQuotedEmptyStringCsv	Specifies whether to restore the legacy behavior of writing nulls as quoted empty strings in CSV output. When false, Spark writes nulls as unquoted empty strings.	false
spark.maxRemoteBlockSizeFetchToMem	Sets the size threshold above which Spark fetches remote blocks to disk instead of memory. This avoids a single large request consuming too much memory.	200m
spark.emr-serverless.allocation.batch.size	Sets the number of executors to request at once in each round of executor allocation.	20

Property Name	Description	Default Value
spark.sql.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Set to -1 to disable broadcasting.	10MB (-1 for CR.4X with 32 workers)
spark.dynamicAllocation.enabled	Specifies whether to use dynamic resource allocation, which scales the number of executors registered with this application up and down based on the workload.	true
spark.files.fetchFailure.unRegisterOutputOnHost	Specifies whether to unregister all map outputs on a host when a fetch failure occurs. When false, Spark unregisters only the outputs of the specific executor that failed, which reduces unnecessary stage recomputation.	false
spark.io.compression.codec	Sets the codec used to compress internal data such as RDD partitions, event log, broadcast variables, and shuffle outputs. Supported values: lz4, snappy, zstd, gzip.	snappy
spark.sql.session.timeZone	Defines the session time zone for handling timestamps in string literals and Java object conversion. Accepts: • Region-based IDs in area/city format (such as America/Los_Angeles) • Zone offsets in (+/-)HH, (+/-)HH:mm, or (+/-)HH:mm:ss format (such as -08 or +01:00) • UTC or Z as aliases for +00:00	UTC

c. For **Data retention in days**, enter the number of days to keep the data.

d. For **Result format,** choose either **CSV** or **Parquet** as the data format the ML input channel should use.

8. For **Service access**, choose the **Existing service role name** that will be used to access this table or choose **Create and use a new service role**.
9. For **Encryption**, choose the **Encrypt secret with a custom KMS key** to specify your own KMS key and related information. Otherwise, Clean Rooms ML will manage the encryption.
10. (Optional) For **Compute payer**, select the collaboration member who pays for query compute costs.

> **Note**
>
> If there is only one payer candidate for query compute in the collaboration, it defaults to that payer.

11. (Optional) For **Synthetic data generation payer**, select the collaboration member who pays for synthetic data generation costs.

> **Note**
>
> This option appears when the ML input channel uses an analysis template configured for synthetic data output. If there is only one payer candidate for synthetic data generation in the collaboration, it defaults to that payer.

12. Choose **Create ML input channel**.

    It will take a few minutes to create the ML input channel. You can see a list of ML input channels on the **ML models** tab.

> **Note**
>
> After the ML input channel is created, you can't edit it.

API

To create an ML input channel (API)

Run the following code with your specific parameters:

```
import boto3
acr_client = boto3.client('cleanroomsml')

acr_client.create_ml_input_channel(
    name="ml_input_channel_name",
    membershipIdentifier='membership_id',

 configuredModelAlgorithmAssociations=[configured_model_algorithm_association_arn],
    retentionInDays=1,
    inputChannel={
        "dataSource": {
            "protectedQueryInputParameters": {
                "sqlParameters": {
                    "queryString": "select * from table",
                    "computeConfiguration": {
                        "worker": {
                            "type": "CR.1X",
                            "number": 16,
                            "properties": {
                                "spark": {
                                    "spark configuration key": "spark configuration
 value",
                                }
                            }
                        }
                    },
                    "resultFormat": "PARQUET"
                }
            }
        },
        "roleArn": "arn:aws:iam::111122223333:role/role_name"
    }
)
channel_arn = resp['ML Input Channel ARN']
```

Creating a trained model in AWS Clean Rooms ML

Prerequisites:

- An AWS account with access to AWS Clean Rooms

- A collaboration set up in AWS Clean Rooms
- A configured model algorithm associated with the collaboration
- At least one configured ML input channel
- Appropriate permissions to create and manage ML models in the collaboration

After you have associated the configured model algorithm to a collaboration, then created and configured an ML input channel, you are ready to create a trained model. A *trained model* is used by members of a collaboration to jointly analyze their data.

You can create a trained model using the following procedure.

Alternatively, you can use incremental training to improve an existing model with new data, or distributed training to train models across multiple compute instances.

Topics

- Using incremental training in AWS Clean Rooms ML
- Using distributed training in AWS Clean Rooms ML

Console

To create a trained model (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration where you want to create a trained model.
4. After the collaboration opens, choose the **ML models** tab.
5. Under **Custom ML models**, in the **Trained models** section, choose **Create trained model**.
6. On the **Create trained model** page, for **Associated model algorithm**, specify the algorithm.
7. For **Trained model details**, enter the following:
   a. For **Name**, enter a unique name for the model in the collaboration.
   b. (Optional) For **Description**, enter a description of the trained model.

c. For **Training data input mode**, choose one of the following:

- Select **File** if you have a smaller dataset that can fit on the ML storage volume and you prefer traditional file system access for your training script.
- Select **Pipe** for large datasets to stream data directly from S3, avoiding the need to download everything to disk, which can improve training speed and reduce storage requirements.
- Select **FastFile** if you want to combine the benefits of streaming from S3 with file system access, especially for sequentially read data or when dealing with fewer files for faster startup times.

8. For **ML input channel details**, do the following:

a. For **ML input channel**, specify the ML input channel that provides data to the model algorithm.

To add another channel, choose **Add another ML input channel**. You can add up to 19 additional ML input channels.

b. For **Channel name**, enter the name of the ML input channel.

c. For **Amazon S3 data distribution type**, choose one of the following:

- Select **Fully replicated** to give each training instance with a complete copy of your dataset. This works best when your dataset is small enough to fit in memory or when each instance needs access to all data.
- Select **Sharded by S3 key** to divide your dataset across training instances based on S3 keys. Each instance receives about 1/n of the total S3 objects, where 'n' is the number of instances. This works best for large datasets that you want to process in parallel.

> **Note**
>
> Consider your dataset size and training requirements when selecting a distribution type. **Fully replicated** provides complete data access but requires more storage, while **Sharded by S3 key** enables distributed processing of large datasets.

9. For **Maximum training duration**, choose the maximum amount of time you want to train your model.
10. For **Hyperparameters**, specify any algorithm-specific parameters and their intended values. Hyperparameters are specific to the model being trained and are used to fine-tune model training.
11. For **Environment variables**, specify any algorithm-specific variables and their intended values. Environment variables are set in the Docker container.
12. For **Encryption**, to use a custom AWS KMS key, select the **Encrypt secret with a custom KMS key** checkbox.
13. For **EC2 Resource configuration**, specify information about the compute resources that are used for model training.
    a. For **Instance type**, choose the type of instance you want to run.
    b. For **Instance count**, enter the number of instances.
    c. For **Volume size in GB**, enter the ML storage volume size.
14. (Optional) For **Model training payer**, select the collaboration member who pays for model training costs.

    > **Note**
    > If there is only one payer candidate for model training in the collaboration, it defaults to that payer.

15. Choose **Create trained model**.

API

To create a trained model (API)

The member with the ability to train a model starts training by selecting the ML input channel and the model algorithm.

Run the following code with your specific parameters:

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')
```

```
acr_ml_client.create_trained_model(
    membershipIdentifier= 'membership_id',
    configuredModelAlgorithmAssociationArn = 'arn:aws:cleanrooms-
ml:region:account:membership/membershipIdentifier/configured-model-algorithm-
association/identifier',
    name='trained_model_name',
    resourceConfig={
        'instanceType': "ml.m5.xlarge",
        'volumeSizeInGB': 1
    },
    dataChannels=[
        {
            "mlInputChannelArn": channel_arn_1,
            "channelName": "channel_name"
        },
        {
            "mlInputChannelArn": channel_arn_2,
            "channelName": "channel_name"
        }
    ]
)
```

> **Note**
>
> After the trained model is created, you can't edit it. To make changes, delete the trained model and create a new one.

Using incremental training in AWS Clean Rooms ML

Prerequisites:

- An AWS account with access to AWS Clean Rooms
- An existing trained model in a collaboration
- A new or updated dataset for incremental training
- Appropriate permissions to create and manage ML models in the collaboration
- Familiarity with the existing model's hyperparameters and configuration

With incremental training, you can use the artifacts of an existing model and an updated dataset to train a new model. Incremental training saves both time and resources.

Use incremental training to:

- Train a new model using an expanded dataset that has an underlying pattern that wasn't accounted for in the earlier training.
- Train several variants of a model, either with different hyperparameters or using different datasets.

Console

To run an incremental training job (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration in which the model artifacts you want to use for incremental training exist.
4. After the collaboration opens, choose the **ML models** tab.
5. Under **Custom ML models**, in the **Trained models** section, choose the radio button next to the trained model you want to incrementally train.
6. On the **Overview** page, under **Versions**,
   a. Choose the radio button next to the trained model you want to incrementally train.
   b. Choose **Train from version**.
7. On the **Create trained model from version** page, for **Trained model version**, choose the version.

   The base model version is automatically selected. You can change this version if other versions exist.
8. For **Trained model details**, enter the following:
   a. For **Name**, enter a unique name for the model in the collaboration.
   b. (Optional) For **Description**, enter a description of the trained model.
   c. For **Training data input mode**, choose one of the following:

- Select **File** if you have a smaller dataset that can fit on the ML storage volume and you prefer traditional file system access for your training script.
- Select **Pipe** for large datasets to stream data directly from S3, avoiding the need to download everything to disk, which can improve training speed and reduce storage requirements.
- Select **FastFile** if you want to combine the benefits of streaming from S3 with file system access, especially for sequentially read data or when dealing with fewer files for faster startup times.

d. For **Incremental training channel name**, enter a name for the incremental training channel

> **Note**
>
> If you specify the **Incremental training channel name** without a version ID, the system uses the base model for incremental training.

9. For **ML input channel details**, do the following:

   a. For **ML input channel**, specify the ML input channel that provides data to the model algorithm.

      To add another channel, choose **Add another ML input channel**. You can add up to 19 additional ML input channels.

   b. For **Channel name**, enter the name of the ML input channel.

   c. For **Amazon S3 data distribution type**, choose one of the following:

      - Select **Fully replicated** to give each training instance with a complete copy of your dataset. This works best when your dataset is small enough to fit in memory or when each instance needs access to all data.
      - Select **Sharded by S3 key** to divide your dataset across training instances based on S3 keys. Each instance receives about 1/n of the total S3 objects, where 'n' is the number of instances. This works best for large datasets that you want to process in parallel.

> **Note**
>
> Consider your dataset size and training requirements when selecting a distribution type. **Fully replicated** provides complete data access but requires more storage, while **Sharded by S3 key** enables distributed processing of large datasets.

10. For **Maximum training duration**, choose the maximum amount of time you want to train your model.
11. For **Hyperparameters**, specify any algorithm-specific parameters and their intended values. Hyperparameters are specific to the model being trained and are used to fine-tune model training.
12. For **Environment variables**, specify any algorithm-specific variables and their intended values. Environment variables are set in the Docker container.
13. For **Encryption**, to use a custom AWS KMS key, select the **Encrypt secret with a custom KMS key** checkbox.
14. For **EC2 Resource configuration**, specify information about the compute resources that are used for model training.
    a. For **Instance type**, choose the type of instance you want to run.
    b. For **Instance count**, enter the number of instances.
    c. For **Volume size in GB**, enter the ML storage volume size.
15. Choose **Create trained model from version**.

API

To run an incremental training job (API)

Run the following code with your specific parameters:

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.create_trained_model(
    membershipIdentifier= 'membership_id',
```

```
    configuredModelAlgorithmAssociationArn = 'arn:aws:cleanrooms-
ml:region:account:membership/membershipIdentifier/configured-model-algorithm-
association/identifier',
    name='trained_model_name',
    resourceConfig={
        'instanceType': 'ml.m5.xlarge',
        'volumeSizeInGB': 1
    },
    incrementalTrainingDataChannels=[
        {
            'trainedModelArn': trained_model_arn,
            'channelName': 'channel_name'
        },
    ]
    dataChannels=[
        {
            'mlInputChannelArn': channel_arn_1,
            'channelName': 'channel_name'
        },
        {
            'mlInputChannelArn': channel_arn_2,
            'channelName': 'channel_name'
        }
    ]
)
```

> **Note**
>
> Limit: Maximum of 20 channels total (including both `dataChannels` and `incrementalTrainingDataChannels`).

> **Note**
>
> After the trained model is created, you can't edit it. To make changes, delete the trained model and create a new one.

Using distributed training in AWS Clean Rooms ML

Prerequisites:

- An AWS account with access to AWS Clean Rooms
- A collaboration set up in AWS Clean Rooms
- A configured model algorithm that supports distributed training
- A large dataset suitable for distributed processing
- Appropriate permissions to create and manage ML models in the collaboration
- Sufficient Amazon EC2 quota to run multiple instances for distributed training

Distributed training leverages the power of many computing nodes working in parallel to process large amounts of data and update model parameters efficiently.

For more information about distributed training, see Distributed training concepts in the *Amazon SageMaker AI Developer Guide*.

Console

To run a distributed training job (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration where you want to create a trained model.
4. After the collaboration opens, choose the **ML models** tab.
5. Under **Custom ML models**, in the **Trained models** section, choose **Create trained model**.
6. On the **Create trained model** page, for **Associated model algorithm**, specify the algorithm
7. For **Trained model details**, enter the following:
   a. For **Name**, enter a unique name for the model in the collaboration.
   b. (Optional) For **Description**, enter a description of the trained model.
   c. For **Training data input mode**, choose one of the following:
      - Select **File** if you have a smaller dataset that can fit on the ML storage volume and you prefer traditional file system access for your training script.

- Select **Pipe** for large datasets to stream data directly from S3, avoiding the need to download everything to disk, which can improve training speed and reduce storage requirements.
- Select **FastFile** if you want to combine the benefits of streaming from S3 with file system access, especially for sequentially read data or when dealing with fewer files for faster startup times.

8. For **ML input channel details**, do the following:

   a. For **ML input channel**, specify the ML input channel that provides data to the model algorithm.

      To add another channel, choose **Add another ML input channel**. You can add up to 19 additional ML input channels.

   b. For **Channel name**, enter the name of the ML input channel.

   c. For **Amazon S3 data distribution type**, choose one of the following:

      - Select **Fully replicated** to give each training instance with a complete copy of your dataset. This works best when your dataset is small enough to fit in memory or when each instance needs access to all data.
      - Select **Sharded by S3 key** to divide your dataset across training instances based on S3 keys. Each instance receives about 1/n of the total S3 objects, where 'n' is the number of instances. This works best for large datasets that you want to process in parallel.

      > **Note**
      >
      > Consider your dataset size and training requirements when selecting a distribution type. **Fully replicated** provides complete data access but requires more storage, while **Sharded by S3 key** enables distributed processing of large datasets.

9. For **Maximum training duration**, choose the maximum amount of time you want to train your model.
10. For **Hyperparameters**, specify any algorithm-specific parameters and their intended values. Hyperparameters are specific to the model being trained and are used to fine-tune model training.

11. For **Environment variables**, specify any algorithm-specific variables and their intended values. Environment variables are set in the Docker container.
12. For **Encryption**, to use a custom AWS KMS key, select the **Encrypt secret with a custom KMS key** checkbox.
13. For **EC2 Resource configuration**, specify information about the compute resources that are used for model training.
    a. For **Instance type**, choose the type of instance you want to run.

       Supported instance types for distributed training are:

       - ml.m5.4xlarge
       - ml.m5.12xlarge
       - ml.m5.2xlarge
       - ml.g5.12xlarge
       - ml.g5.24xlarge

    b. For **Instance count**, enter the number of instances.
    c. For **Volume size in GB**, enter the ML storage volume size.
14. Choose **Create trained model**.

API

To run a distributed training job (API)

Run the following code with your specific parameters:

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.create_trained_model(
    membershipIdentifier= 'membership_id',
    configuredModelAlgorithmAssociationArn = 'arn:aws:cleanrooms-
ml:region:account:membership/membershipIdentifier/configured-model-algorithm-
association/identifier',
    name='trained_model_name',
    trainingInputMode: "File",
    resourceConfig={
        'instanceCount': "3"
```

```
        'instanceType': "ml.m5.xlarge",
        'volumeSizeInGB': 3
    },
    dataChannels=[
        {
            "mlInputChannelArn": channel_arn_1,
            "channelName": "channel_name",
            "S3DataDistributionType:" "FullyReplicated"
        }

    ]
)
```

> **Note**
>
> After the trained model is created, you can't edit it. To make changes, delete the trained model and create a new one.

Exporting model artifacts from AWS Clean Rooms ML

This task is optional and should be completed when you have assigned the `CAN_RECEIVE_MODEL_OUTPUT` member ability to a member of the collaboration.

After model training has completed, the member who trained the model can initiate the export of model artifacts. The member who trained the model chooses who will receive model artifacts, provided that member can receive results and a valid ML configuration.

Console

To configure a custom ML model algorithm (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration that contains the custom model that you want to export.
4. After the collaboration opens, choose the **ML Models** tab, then choose your model from the **Custom trained model** table

5. On the custom trained model details page, click **Export model output**.
6. For **Export model output**, for **Export model output details**, enter a **Name** and optional **Description**.

   Choose which member will receive the model artifacts in the **Model output exported to members of collaboration** drop down list.
7. Choose **Export**.

   The results are exported to the following path in the Amazon S3 location that was specified in the ML configuration: `yourSpecifiedS3Path/collaborationIdentifier/trainedModelName/callerAccountId/jobName`. Only the **Files to export**, up to the maximum file size specified, that you selected when associating the configured model algorithm are exported.

API

To configure a custom ML model algorithm (API)

Initiate the model export by running the following code:

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.start_trained_model_export_job(
    membershipIdentifier='membership_id',
    trainedModelArn='arn:aws:cleanrooms-ml:region:account:membership/
membershipIdentifier/trained-model/identifier',
    outputConfiguration={
        'member': {
            'accountId': 'model_output_receiver_account'
        }
    },
    name='export_job_name'
)
```

The results are exported to the following path in the Amazon S3 location that was specified in the ML configuration: `yourSpecifiedS3Path/collaborationIdentifier/trainedModelName/callerAccountId/jobName`. Only the `filesToExport`, up to the `maxSize` specified, that you selected when associating the configured model algorithm are exported.

Running inference on a trained model in AWS Clean Rooms ML

Members with the ability to run queries can also initiate inference job once the training job is complete. They pick the inference dataset against which they want to run inference and reference the trained model outputs that they'd like to run inference container with.

The member who will receive inference output must be granted the member ability `CAN_RECEIVE_INFERENCE_OUTPUT`.

Console

To create a model inference job (console)

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console at https://console.aws.amazon.com/cleanrooms.
2. In the left navigation pane, choose **Collaborations**.
3. On the **Collaborations** page, choose the collaboration that contains the custom model that you want to create an inference job on.
4. After the collaboration opens, choose the **ML Models** tab, then choose your model from the **Custom trained model** table.
5. On the custom trained model details page, click **Start inference job**.
6. For **Start inference job**, for **Inference job details**, enter a **Name** and optional **Description**.

   Enter the following information:

   - **Associated model algorithm** – The associated model algorithm that is used during the inference job.
   - **ML input channel details** – The ML input channel that will provide the data for this inference job.
   - **Transform resources** – The compute instance that is used to perform the transform function of the inference job.
   - **Output configuration** – Who will receive the inference job output and the MIME type of the output.
   - **Encryption** – Choose the **Customize encryption settings** to specify your own KMS key and related information. Otherwise, Clean Rooms ML will manage the encryption.
   - **Transform job details** – The maximum payload of the inference job, in MB.

- **Environment variables** – Any environment variables necessary to access the inference job container image.

7. (Optional) For **Model inference payer**, select the collaboration member who pays for model inference costs.

> **Note**
>
> If there is only one payer candidate for model inference in the collaboration, it defaults to that payer.

8. Choose **Start inference job**.

   The results are exported to the following path in the Amazon S3 location that was specified in the ML configuration: `yourSpecifiedS3Path/collaborationIdentifier/trainedModelName/callerAccountId/jobName`.

API

To create a model inference job (API)

Initiate the inference job by running the following code:

```
import boto3
acr_ml_client= boto3.client('cleanroomsml')

acr_ml_client.start_trained_model_inference_job(
    name="inference_job",
    membershipIdentifier='membership_id',
    trainedModelArn='arn:aws:cleanrooms-ml:region:account:membership/
membershipIdentifier/trained-model/identifier',

    dataSource={
        "mlInputChannelArn": 'channel_arn_3'
    },
    resourceConfig={'instanceType': 'ml.m5.xlarge'},
    outputConfiguration={
        'accept': 'text/csv',
        'members': [
            {
                "accountId": 'member_account_id'
            }
```

```
        ]
    }
)
```

The results are exported to the following path in the Amazon S3 location that was specified in the ML configuration: `yourSpecifiedS3Path/collaborationIdentifier/trainedModelName/callerAccountId/jobName`.

Create AWS Clean Rooms ML models as a training data provider

A *lookalike model* is a model of a training data provider's data that allows a seed data provider to create a lookalike segment of training data provider's data that most closely resembles their seed data. To create a lookalike model that can be used in a collaboration, you must import your training data, create a lookalike model, configure that lookalike model, and then associate it to a collaboration.

Working with lookalike models requires that two parties, a training data provider and a seed data provider, work sequentially in AWS Clean Rooms to bring their data into a collaboration. This is the workflow that the training data provider must complete first:

1. The training data provider's data must be stored in a AWS Glue data catalog table of user-item interactions. At a minimum, the training data must contain a user ID column, interaction ID column, and a timestamp column.
2. The training data provider registers the training data with AWS Clean Rooms.
3. The training data provider creates a lookalike model that can be shared with multiple seed data providers. The lookalike model is a deep neural network that can take up to 24 hours to train. It isn't automatically retrained and we recommend that you retrain the model weekly.
4. The training data provider configures the lookalike model, including whether to share relevance metrics and the Amazon S3 location of the output segments. The training data provider can create multiple configured lookalike models from a single lookalike model.
5. The training data provider associates the configured audience model to a collaboration that's shared with a seed data provider.

After the training data provider is done creating the ML model, the seed data provider can create and export the lookalike segment.

Topics

- Importing training data
- Creating a lookalike model
- Configuring a lookalike model
- Associating a configured lookalike model

- Updating a configured lookalike model

Importing training data

> **Note**
>
> You can only supply a training data set for using in a Clean Rooms ML lookalike model that has data stored in Amazon S3. However, you can supply the seed data for a lookalike model using SQL that runs across data stored in any supported data source.

Before you create a lookalike model, you must specify the AWS Glue table that contains the training data. Clean Rooms ML doesn't store a copy of this data, just metadata that allows it to access the data.

To import training data in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **AWS ML models**.
3. On the **Training datasets** tab, choose **Create training dataset**.
4. On the **Create training dataset** page, for **Training dataset details**, enter a **Name** and optional **Description**.
5. Choose the **Training data source** by selecting the **Database** and **Table** that you want to configure from the dropdown lists.

   > **Note**
   >
   > To verify that this is the correct table, do either one of the following:
   >
   > - Choose **View in AWS Glue**.
   > - Turn on **View schema** to view the schema.

6. For **Training details**, choose the **User identifier column**, **Item identifier column**, and **Timestamp column** from the dropdown lists. The training data must contain these three columns. You can also select any other columns that you want to include in the training data.

The data in the **Timestamp column** must be in the Unix epoch time in seconds format.

7. (Optional) If you have any **Additional columns to train**, choose the **Column name** and **Type** from the dropdown lists.
8. In **Service access**, you must specify a service role that can access your data and provide a KMS key if your data is encrypted. Choose **Create and use a new service role** and Clean Rooms ML will automatically create a service role and add the necessary permissions policy. Choose **Use an existing service role** and enter it in the **Service role name** field if you have a specific service role that you want to use.

   If your data is encrypted, enter your KMS key in the **AWS KMS key** field, or click **Create an AWS KMS key** to generate a new KMS key.
9. If you want to enable **Tags** for the training dataset, choose **Add new tag** and then enter the **Key** and **Value** pair.
10. Choose **Create training dataset**.

For the corresponding API action, see CreateTrainingDataset.

Creating a lookalike model

After you have created a training dataset, you are ready to create a lookalike model. You can create many lookalike models from a single training dataset.

You must create a default database in your AWS Glue Data Catalog or include the `glue:createDatabase` permission in the provided role.

To create a lookalike model in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **AWS ML models**.
3. On the **Lookalike models** tab, choose **Create lookalike model**.
4. On the **Create lookalike model** page, for **Lookalike model details**, enter a **Name** and optional **Description**.

   a. Choose the **Training dataset** that you want to model from the dropdown list.

> **Note**
>
> To verify that this is the correct training dataset, turn on **Show training dataset details** to view the details.
> To create a new training dataset, choose **Create training dataset**.

b. (Optional) Enter a **Training window**.

5. If you want to enable custom encryption settings for the lookalike model, choose **Customize encryption settings** and then enter the KMS key.
6. If you want to enable **Tags** for the lookalike model, choose **Add new tag** and then enter the **Key** and **Value** pair.
7. Choose **Create lookalike model**.

> **Note**
>
> Model training can take several hours to 2 days.

For the corresponding API action, see CreateAudienceModel.

Configuring a lookalike model

After you have created a lookalike model, you are ready to configure it for use in a collaboration. You can create multiple configured lookalike models from a single lookalike model.

To configure a lookalike model in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **AWS ML models**.
3. On the **Configured lookalike models** tab, choose **Configure lookalike model**.
4. On the **Configure lookalike model** page, for **Configured lookalike model details**, enter a **Name** and optional **Description**.

   a. Choose the **Lookalike model** that you want to configure from the dropdown list.

> **Note**
>
> To verify that this is the correct lookalike model, turn on **Show lookalike model details** to view the details.
> To create a new lookalike model, choose **Create lookalike model**.

b. Choose the **Minimum matching seed size** that you want. This is the minimum number of users in the seed data provider's data that overlap with users in the training data. This value must be greater than 0.

5. For **Metrics to share with other members**, choose whether you want the seed data provider in your collaboration to receive model metrics, including relevance scores.
6. For **Lookalike segment destination location**, enter the Amazon S3 bucket where the lookalike segment is exported. This bucket must be located in the same region as your other resources.
7. For **Service access**, choose the **Existing service role name** that will be used to access this table.
8. For **Advanced bin size configuration**, specify the **Audience size type** as either an **Absolute** number or a **Percentage**.
9. If you want to enable **Tags** for the configured table resource, choose **Add new tag** and then enter the **Key** and **Value** pair.
10. Choose **Configure lookalike model**.

For the corresponding API action, see CreateConfiguredAudienceModel.

Associating a configured lookalike model

After you have configured a lookalike model, you can associate it to a collaboration.

To associate a configured lookalike model in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. On the **With active membership** tab, choose a collaboration.
4. On the **ML models** tab, under **Ready-to-use lookalike models**, choose **Associate lookalike model**.

5. On the **Associate configured lookalike model** page, for **Configured lookalike model association details**:

   a. Enter a **Name** for the associated configured audience model.

   b. Enter a **Description** of the table.

      The description helps differentiate between other associated configured audience models with similar names.

6. For **Configured lookalike model**, choose a configured lookalike model from the dropdown list.
7. Choose **Associate**.

For the corresponding API action, see CreateConfiguredAudienceModelAssociation.

Updating a configured lookalike model

After you have associated a configured a lookalike model, you can update it to change information such as the name, metrics to share, or output Amazon S3 location.

To update an associated configured lookalike model in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **AWS ML models**.
3. On the **Configured lookalike models** tab, under **Ready-to-use lookalike models**, choose a configured lookalike model and select **Edit**.
4. On the **Edit** page, for **Configured lookalike model association details**:

   a. Update the **Name** and optional **Description**.

   b. Choose the **Lookalike model** that you want configured from the dropdown list.

   c. Choose the **Minimum matching seed size** that you want. This is the minimum number of users in the seed data provider's data that overlap with users in the training data. This value must be greater than 0.

5. For **Metrics to share with other members**, choose whether you want the seed data provider in your collaboration to receive model metrics, including relevance scores.
6. For **Lookalike segment destination location**, enter the Amazon S3 bucket where lookalike segment is exported. This bucket must be located in the same region as your other resources.

7. For **Service access**, choose the **Existing service role name** that will be used to access this table.
8. For **Advanced bin size configuration**, choose how you want to configure the audience bin sizes.
9. Choose **Save changes**.

For the corresponding API action, see UpdateConfiguredAudienceModel.

Creating AWS Clean Rooms ML models as a seed data provider

After the training data provider is done creating the ML model, the seed data provider can create and export the lookalike segment. The lookalike segment is a subset of the training data that most closely resembles the seed data.

This is the workflow that the seed data provider must complete:

1. The seed data provider's data can be stored in an Amazon S3 bucket or it can come from the results of query.
2. The seed data provider opens the collaboration that they share with the training data provider.
3. The seed data provider creates a lookalike segment from the Clean Rooms ML tab of the collaboration page.
4. The seed data provider can evaluate the relevance metrics, if they were shared, and export the lookalike segment for use outside AWS Clean Rooms.

Topics

- Creating a lookalike segment
- Exporting a lookalike segment

Creating a lookalike segment

> **Note**
>
> You can only supply a training data set for using in a Clean Rooms ML lookalike model that has data stored in Amazon S3. However, you can supply the seed data for a lookalike model using SQL that runs across data stored in any supported data source.

A lookalike segment is a subset of the training data that most closely resembles the seed data.

To create a lookalike segment in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. On the **With active membership** tab, choose a collaboration.
4. On the **ML Models** tab, choose **Create lookalike segment**.
5. On the **Create lookalike segment** page, for **Associated configured lookalike model**, choose the associated configured lookalike model to use for this lookalike segment.
6. For **Lookalike segment details** enter a **Name** and optional **Description**.
7. For **Seed profiles**, choose your **Seed method** by selecting an option and then taking the recommended action.

Option	Recommended action
Amazon S3 path	1. Select an Amazon S3 location. 2. (Optional) Choose **Include seed profiles in the output**.
SQL query	Write a SQL query and use its results as the seed data.
Analysis template	Choose an analysis template from the dropdown list and use the results created by an analysis template.

8. Choose the **Worker type** to use when creating this data source. The default worker type is **CR.1X**. Specify the **Number of workers** to use. The default is worker number **16**. To specify **Spark properties**:
   a. Expand **Spark properties**.
   b. Choose **Add Spark properties**.
   c. On the **Spark properties** dialog box, choose a **Property name** from the dropdown list and enter a **Value**.

The following tables provide a definition for each property.

For more information about Spark properties, see Spark Properties in the Apache Spark documentation.

> **Note**
>
> You can configure a maximum of 50 Spark properties. Each property value can be up to 500 characters.

Property Name	Description	Default Value
spark.task.maxFailures	Controls how many consecutive times a task can fail before the job fails. Requires a value greater than or equal to 1. The number of allowed retries equals this value minus 1. The failure count resets if any attempt succeeds. Failures across different tasks don't accumulate toward this limit.	4
spark.sql.files.maxPartitionBytes	Sets the maximum number of bytes to pack into a single partition when reading from file-based sources such as Parquet, JSON, and ORC.	128MB
spark.hadoop.fs.s3.maxRetries	Sets the maximum number of retry attempts for Amazon S3 file operations.	(none)

Property Name	Description	Default Value
spark.network.timeout	Sets the default timeout for all network interactions. Overrides the following timeout settings if they aren't configured: • spark.storage.blockManagerHeartbeatTimeoutMs • spark.shuffle.io.connectionTimeout • spark.rpc.askTimeout • spark.rpc.lookupTimeout	120s
spark.rdd.compress	Specifies whether to compress serialized RDD partitions using spark.io.compression.codec. Applies to StorageLevel.MEMORY_ONLY_SER in Java and Scala, or StorageLevel.MEMORY_ONLY in Python. Reduces storage space but requires additional CPU processing time.	false
spark.shuffle.spill.compress	Specifies whether to compress shuffle spill data using spark.io.compression.codec.	true
spark.shuffle.compress	Specifies whether to compress map output files. Compression uses spark.io.compression.codec.	true
spark.shuffle.service.index.cache.size	Sets the cache size limit, in bytes unless otherwise specified.	100m
spark.shuffle.io.maxRetries	Sets the maximum number of retries for fetches that fail due to IO-related exceptions.	3
spark.shuffle.io.retryWait	Sets the wait time between retries of fetches. The maximum delay caused by retrying is 15 seconds by default, calculated as maxRetries * retryWait.	5s

Property Name	Description	Default Value
spark.shuffle.io.connectionTimeout	Sets the timeout for established connections between shuffle servers and clients to be marked as idle and closed if there are still outstanding fetch requests but no traffic on the channel.	(value of spark.network.timeout)
spark.driver.maxResultSize	Sets the total size limit of serialized results of all partitions for each Spark action, in bytes. Should be at least 1M, or 0 for unlimited.	1g
spark.memory.fraction	Sets the fraction of (heap space - 300MB) used for execution and storage. The lower this value, the more frequently spills and cached data eviction occur. Leaving this at the default value is recommended.	0.6
spark.scheduler.mode	Sets the scheduling mode between jobs submitted to the same SparkContext. Can be set to FAIR to use fair sharing instead of queueing jobs one after another. Supported values: FAIR, FIFO.	FIFO
spark.sql.adaptive.advisoryPartitionSizeInBytes	Sets the target size in bytes for shuffle partitions during adaptive optimization when spark.sql.adaptive.enabled is true. Controls partition size when coalescing small partitions or splitting skewed partitions.	(value of spark.sql.adaptive.shuffle.targetPostShuffleInputSize)
spark.sql.adaptive.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Applies only in adaptive framework. Uses the same default value as spark.sql.autoBroadcastJoinThreshold. Set to -1 to disable broadcasting.	(none)

Property Name	Description	Default Value
spark.sql.adaptive.coalescePartitions.enabled	Specifies whether to coalesce contiguous shuffle partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes to optimize task size. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Defines the initial number of shuffle partitions before coalescing. Requires both spark.sql.adaptive.enabled and spark.sql.adaptive.coalescePartitions.enabled to be true. Defaults to the value of spark.sql.shuffle.partitions.	(none)
spark.sql.adaptive.coalescePartitions.minPartitionSize	Sets the minimum size for coalesced shuffle partitions to prevent partitions from becoming too small during adaptive optimization.	1 MB
spark.sql.adaptive.coalescePartitions.parallelismFirst	Specifies whether to calculate partition sizes based on cluster parallelism instead of spark.sql.adaptive.advisoryPartitionSizeInBytes during partition coalescing. Generates smaller partition sizes than the configured target size to maximize parallelism. We recommend setting this to false on busy clusters to improve resource utilization by preventing excessive small tasks.	true
spark.sql.adaptive.enabled	Specifies whether to enable adaptive query execution to re-optimize query plans during query execution, based on accurate runtime statistics.	true
spark.sql.adaptive.forceOptimizeSkewedJoin	Specifies whether to force enable OptimizeSkewedJoin even if it introduces extra shuffle.	false

Property Name	Description	Default Value
spark.sql.adaptive.localShuffleReader.enabled	Specifies whether to use local shuffle readers when shuffle partitioning isn't required, such as after converting from sort-merge joins to broadcast-hash joins. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Sets the maximum partition size in bytes for building local hash maps. Prioritizes shuffled hash joins over sort-merge joins when: • This value equals or exceeds spark.sql.adaptive.advisoryPartitionSizeInBytes • All partition sizes are within this limit Overrides spark.sql.join.preferSortMergeJoin setting.	0 bytes
spark.sql.adaptive.optimizeSkewsInRebalancePartitions.enabled	Specifies whether to optimize skewed shuffle partitions by splitting them into smaller partitions based on spark.sql.adaptive.advisoryPartitionSizeInBytes. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.rebalancePartitionsSmallPartitionFactor	Defines the size threshold factor for merging partitions during splitting. Partitions smaller than this factor multiplied by spark.sql.adaptive.advisoryPartitionSizeInBytes are merged.	0.2

Property Name	Description	Default Value
spark.sql.adaptive.skewJoin.enabled	Specifies whether to handle data skew in shuffled joins by splitting and optionally replicating skewed partitions. Applies to sort-merge and shuffled hash joins. Requires spark.sql.adaptive.enabled to be true.	true
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Determines the size factor that determines partition skew. A partition is skewed when its size exceeds both: • This factor multiplied by the median partition size • The value of spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Sets the size threshold in bytes for identifying skewed partitions. A partition is skewed when its size exceeds both: • This threshold • The median partition size multiplied by spark.sql.adaptive.skewJoin.skewedPartitionFactor We recommend setting this value larger than spark.sql.adaptive.advisoryPartitionSizeInBytes.	256MB
spark.sql.broadcastTimeout	Controls the timeout period in seconds for the broadcast operations during broadcast joins.	300 seconds
spark.sql.cbo.enabled	Specifies whether to enable cost-based optimization (CBO) for plan statistics estimation.	false

Property Name	Description	Default Value
spark.sql.cbo.joinReorder.dp.star.filter	Specifies whether to apply star-join filter heuristics during cost-based join enumeration.	false
spark.sql.cbo.joinReorder.dp.threshold	Sets the maximum number of joined nodes allowed in the dynamic programming algorithm.	12
spark.sql.cbo.joinReorder.enabled	Specifies whether to enable join reordering in cost-based optimization (CBO).	false
spark.sql.cbo.planStats.enabled	Specifies whether to fetch row counts and column statistics from the catalog during logical plan generation.	false
spark.sql.cbo.starSchemaDetection	Specifies whether to enable join reordering based on star schema detection.	false
spark.sql.files.maxPartitionNum	Sets the target maximum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Rescales partitions when the initial count exceeds this value. This is a suggested target, not a guaranteed limit.	(none)
spark.sql.files.maxRecordsPerFile	Sets the maximum number of records to write to a single file. No limit applies when set to zero or a negative value.	0
spark.sql.files.minPartitionNum	Sets the target minimum number of split file partitions for file-based sources (Parquet, JSON, and ORC). Defaults to spark.sql.leafNodeDefaultParallelism. This is a suggested target, not a guaranteed limit.	(none)

Property Name	Description	Default Value
spark.sql.inMemoryColumnarStorage.batchSize	Controls the batch size for columnar caching. Increasing the size improves memory utilization and compression but increases the risk of out-of-memory errors.	10000
spark.sql.inMemoryColumnarStorage.compressed	Specifies whether to automatically select compression codecs for columns based on data statistics.	true
spark.sql.inMemoryColumnarStorage.enableVectorizedReader	Specifies whether to enable vectorized reading for columnar caching.	true
spark.sql.legacy.allowHashOnMapType	Specifies whether to allow hash operations on map type data structures. This legacy setting maintains compatibility with older Spark versions' map type handling.	(none)
spark.sql.legacy.allowNegativeScaleOfDecimal	Specifies whether to allow negative scale values in decimal type definitions. This legacy setting maintains compatibility with older Spark versions that supported negative decimal scales.	(none)
spark.sql.legacy.castComplexTypesToString.enabled	Specifies whether to enable legacy behavior for casting complex types to strings. Maintains compatibility with older Spark versions' type conversion rules.	(none)
spark.sql.legacy.charVarcharAsString	Specifies whether to treat CHAR and VARCHAR types as STRING types. This legacy setting provides compatibility with older Spark versions' string type handling.	(none)

Property Name	Description	Default Value
spark.sql.legacy.createEmptyCollectionUsingStringType	Specifies whether to create empty collections using string type elements. This legacy setting maintains compatibility with older Spark versions' collection initialization behavior.	(none)
spark.sql.legacy.exponentLiteralAsDecimal.enabled	Specifies whether to interpret exponential literals as decimal types. This legacy setting maintains compatibility with older Spark versions' numeric literal handling.	(none)
spark.sql.legacy.json.allowEmptyString.enabled	Specifies whether to allow empty strings in JSON processing. This legacy setting maintains compatibility with older Spark versions' JSON parsing behavior.	(none)
spark.sql.legacy.parquet.int96RebaseModeInRead	Specifies whether to use legacy INT96 timestamp rebase mode when reading Parquet files. This legacy setting maintains compatibility with older Spark versions' timestamp handling.	(none)
spark.sql.legacy.timeParserPolicy	Controls the time parsing behavior for backwards compatibility. This legacy setting determines how timestamps and dates are parsed from strings.	(none)
spark.sql.legacy.typeCoercion.datetimeToString.enabled	Specifies whether to enable legacy type coercion behavior when converting datetime values to strings. Maintains compatibility with older Spark versions' datetime conversion rules.	(none)
spark.sql.maxSinglePartitionBytes	Sets the maximum partition size in bytes. The planner introduces shuffle operations for larger partitions to improve parallelism.	128m

Property Name	Description	Default Value
spark.sql.metadataCacheTTLSeconds	Controls the time-to-live (TTL) for metadata caches. Applies to partition file metadata and session catalog caches. Requires: • A positive value greater than zero • spark.sql.catalogImplementation set to hive • spark.sql.hive.filesourcePartitionFileCacheSize greater than zero • spark.sql.hive.manageFilesourcePartitions set to true	-1000ms
spark.sql.optimizer.collapseProjectAlwaysInline	Specifies whether to collapse adjacent projections and inline expressions, even when it causes duplication.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	Specifies whether to generate predicates for partition columns used as join keys.	true
spark.sql.optimizer.enableCsvExpressionOptimization	Specifies whether to optimize CSV expressions in SQL optimizer by pruning unnecessary columns from from_csv operations.	true
spark.sql.optimizer.enableJsonExpressionOptimization	Specifies whether to optimize JSON expressions in SQL optimizer by: • Pruning unnecessary columns from from_json operations • Simplifying from_json and to_json combinations • Optimizing named_struct operations	true

Property Name	Description	Default Value
spark.sql.optimizer.excludedRules	Defines optimizer rules to disable, identified by comma-separated rule names. Some rules cannot be disabled as they are required for correctness. The optimizer logs which rules are successfully disabled.	(none)
spark.sql.optimizer.runtime.bloomFilter.applicationSideScanSizeThreshold	Sets the minimum aggregated scan size in bytes required to inject a Bloom filter on the application side.	10GB
spark.sql.optimizer.runtime.bloomFilter.creationSideThreshold	Defines the maximum size threshold for injecting a Bloom filter on the creation side.	10MB
spark.sql.optimizer.runtime.bloomFilter.enabled	Specifies whether to insert a Bloom filter to reduce shuffle data when one side of a shuffle join has a selective predicate.	true
spark.sql.optimizer.runtime.bloomFilter.expectedNumItems	Defines the default number of expected items in the runtime Bloom filter.	1000000
spark.sql.optimizer.runtime.bloomFilter.maxNumBits	Sets the maximum number of bits allowed in the runtime Bloom filter.	67108864
spark.sql.optimizer.runtime.bloomFilter.maxNumItems	Sets the maximum number of expected items allowed in the runtime Bloom filter.	4000000
spark.sql.optimizer.runtime.bloomFilter.numBits	Defines the default number of bits used in the runtime Bloom filter.	8388608

Property Name	Description	Default Value
spark.sql.optimizer.runtime.rowLevelOperationGroupFilter.enabled	Specifies whether to enable runtime group filtering for row-level operations. Allows data sources to: • Prune entire groups of data (such as files or partitions) using data source filters • Execute runtime queries to identify matching records • Discard unnecessary groups to avoid expensive rewrites Limitations: • Not all expressions can convert to data source filters • Some expressions require Spark evaluation (such as subqueries)	true
spark.sql.optimizer.runtimeFilter.number.threshold	Sets the total number of injected runtime filters (non-DPP). This is to prevent driver OOMs with too many Bloom filters.	10
spark.sql.optimizer.runtimeFilter.semiJoinReduction.enabled	Specifies whether to insert a semi-join to reduce shuffle data when one side of a shuffle join has a selective predicate.	false

Property Name	Description	Default Value
spark.sql.parquet.aggregatePushdown	Specifies whether to push down aggregates to Parquet for optimization. Supports: • MIN and MAX for boolean, integer, float, and date types • COUNT for all data types Throws an exception if statistics are missing from any Parquet file footer.	false
spark.sql.parquet.columnarReaderBatchSize	Controls the number of rows in each Parquet vectorized reader batch. Choose a value that balances performance overhead and memory usage to prevent out-of-memory errors.	4096
spark.sql.parquet.enableVectorizedReader	Specifies whether to enable vectorized Parquet decoding.	true
spark.sql.shuffle.partitions	Sets the default number of partitions for data shuffling during joins or aggregations. Cannot be modified between structured streaming query restarts from the same checkpoint location.	200
spark.sql.shuffledHashJoinFactor	Defines the multiplication factor used to determine shuffle hash join eligibility. A shuffle hash join is selected when the small-side data size multiplied by this factor is less than the large-side data size.	3

Property Name	Description	Default Value
spark.sql.sources.parallelPartitionDiscovery.threshold	Sets the maximum number of paths for driver-side file listing with file-based sources (Parquet, JSON, and ORC). When exceeded during partition discovery, files are listed using a separate Spark distributed job.	32
spark.sql.statistics.histogram.enabled	Specifies whether to generate equi-height histograms during column statistics computation to improve estimation accuracy. Requires an additional table scan beyond the one needed for basic column statistics.	false
spark.dynamicAllocation.executorIdleTimeout	Sets the duration an executor must be idle before it is removed when dynamic allocation is enabled.	60s
spark.dynamicAllocation.schedulerBacklogTimeout	Sets the duration that pending tasks must be backlogged before new executors are requested when dynamic allocation is enabled.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Same as spark.dynamicAllocation.schedulerBacklogTimeout, but used only for subsequent executor requests.	(value of spark.dynamicAllocation.schedulerBacklogTimeout)

Property Name	Description	Default Value
spark.scheduler.minRegisteredResourcesRatio	Sets the minimum ratio of registered resources (registered resources / total expected resources) to wait for before scheduling begins. Specified as a double between 0.0 and 1.0. Regardless of whether the minimum ratio of resources has been reached, the maximum amount of time it will wait before scheduling begins is controlled by spark.scheduler.maxRegisteredResourcesWaitingTime.	0.8
spark.scheduler.maxRegisteredResourcesWaitingTime	Sets the maximum amount of time to wait for resources to register before scheduling begins.	30s
spark.sql.hive.metastorePartitionPruningFallbackOnException	Specifies whether to fall back to getting all partitions from Hive metastore and perform partition pruning on the Spark client side when encountering MetaException from the metastore.	false
spark.sql.crossJoin.enabled	Specifies whether to allow queries that contain a cartesian product without explicit CROSS JOIN syntax.	true
spark.sql.analyzer.maxIterations	Sets the maximum number of iterations the query analyzer runs before giving up. Higher values allow the analyzer to process very large or deeply nested queries.	100
spark.sql.datapreffetch.filescan.maxParallelismPerTask	Sets the maximum number of file splits to pre-fetch concurrently for each task when scanning files.	4

Property Name	Description	Default Value
spark.sql.iceberg.data-prefetch.enabled	Specifies whether to enable data pre-fetch optimization when reading Iceberg tables.	true
spark.sql.legacy.nullValueWrittenAsQuotedEmptyStringCsv	Specifies whether to restore the legacy behavior of writing nulls as quoted empty strings in CSV output. When false, Spark writes nulls as unquoted empty strings.	false
spark.maxRemoteBlockSizeFetchToMem	Sets the size threshold above which Spark fetches remote blocks to disk instead of memory. This avoids a single large request consuming too much memory.	200m
spark.emr-serverless.allocation.batch.size	Sets the number of executors to request at once in each round of executor allocation.	20

Property Name	Description	Default Value
spark.sql.autoBroadcastJoinThreshold	Sets the maximum table size in bytes for broadcasting to worker nodes during joins. Set to -1 to disable broadcasting.	10MB (-1 for CR.4X with 32 workers)
spark.dynamicAllocation.enabled	Specifies whether to use dynamic resource allocation, which scales the number of executors registered with this application up and down based on the workload.	true

Property Name	Description	Default Value
spark.files.fetchFailure.unRegisterOutputOnHost	Specifies whether to unregister all map outputs on a host when a fetch failure occurs. When false, Spark unregisters only the outputs of the specific executor that failed, which reduces unnecessary stage recomputation.	false
spark.io.compression.codec	Sets the codec used to compress internal data such as RDD partitions, event log, broadcast variables, and shuffle outputs. Supported values: lz4, snappy, zstd, gzip.	snappy
spark.sql.session.timeZone	Defines the session time zone for handling timestamps in string literals and Java object conversion. Accepts: • Region-based IDs in area/city format (such as America/Los_Angeles) • Zone offsets in (+/-)HH, (+/-)HH:mm, or (+/-)HH:mm:ss format (such as -08 or +01:00) • UTC or Z as aliases for +00:00	UTC

9. For **Service access**, choose the **Existing service role name** that will be used to access this table.
10. If you want to enable **Tags** for the training dataset, choose **Add new tag** and then enter the **Key** and **Value** pair.
11. Choose **Create lookalike segment**.

For the corresponding API action, see StartAudienceGenerationJob.

Exporting a lookalike segment

After you have created a lookalike segment, you can export that data to an Amazon S3 bucket.

To export a lookalike segment in AWS Clean Rooms

1. Sign in to the AWS Management Console and open the AWS Clean Rooms console with your AWS account (if you haven't yet done so).
2. In the left navigation pane, choose **Collaborations**.
3. On the **With active membership** tab, choose a collaboration.
4. On the **ML Models** tab, select a lookalike segment and choose **Export**.
5. For **Export lookalike model**, for **Export lookalike model details** enter a **Name** and optional **Description**.
6. For **Segment size**, choose the size you want for the exported segment.
7. Choose **Export**.

For the corresponding API action, see StartAudienceExportJob.

Troubleshooting AWS Clean Rooms

If you encounter issues while using AWS Clean Rooms, you can use the AWS MCP Server to automatically diagnose problems, or refer to the common issues listed in the topics below for manual resolution.

Using AWS MCP to troubleshoot Clean Rooms issues

You can use the AWS MCP Server with any MCP-compatible client to automatically diagnose and resolve common Clean Rooms issues. The aws-cleanrooms skill can help with the following:

- **Permission errors** – Access denied errors when running protected queries, writing results, accessing configured tables, or performing cross-account operations. The skill systematically checks IAM role policies, S3 bucket policies, KMS key policies, Lake Formation permissions, and cross-account trust configuration.
- **Missing CloudWatch logs for custom ML model jobs** – CloudWatch logs not appearing for custom model training or inference jobs. The skill checks privacy configuration, ML Configuration setup, role permissions, and log group existence.

To get started, install and configure the AWS MCP server. For setup instructions, see the AWS MCP Server Setup guide.

Common issues

This section describes some common issues that might arise when using AWS Clean Rooms and how to fix them.

Issues

- One or more tables referenced by the query is not accessible by its associated service role. The table/role owner must grant the service role access to the table.
- One of the underlying datasets has an unsupported file format.
- Query results are not as expected when using Cryptographic Computing for Clean Rooms.
- AWS Clean Rooms Spark SQL: Missing partition data

One or more tables referenced by the query is not accessible by its associated service role. The table/role owner must grant the service role access to the table.

- Verify that the permissions for the service role are set up as required. For more information, see Setting up AWS Clean Rooms.

One of the underlying datasets has an unsupported file format.

- Ensure that your dataset is in one of the supported file formats:
  - Parquet
  - RCFile
  - TextFile
  - SequenceFile
  - RegexSerde
  - OpenCSV
  - AVRO
  - JSON

  For more information, see Data formats for AWS Clean Rooms.

Query results are not as expected when using Cryptographic Computing for Clean Rooms.

If you are using Cryptographic Computing for Clean Rooms (C3R), verify that your query uses encrypted columns correctly:

- The sealed columns are only used in SELECT clauses.
- The fingerprint columns are only used in JOIN clauses (and GROUP BY clauses under certain conditions).
- That you are only JOINing fingerprint columns with the same name if the collaboration settings require it.

For more information, see the section called “Cryptographic computing” and the section called “Column types”.

AWS Clean Rooms Spark SQL: Missing partition data

All partitions in the AWS Glue Data Catalog must also have data in S3. The engine uses the Spark setting `spark.sql.files.ignoreMissingFiles=False`

For more information, see https://spark.apache.org/docs/latest/sql-data-sources-generic-options.html#ignore-missing-files .

If you encounter this error you will get the following error message: `"Missing partition data: One of the configured tables is partitioned and one or more of the partitions does not have data".`

Compare your data present in Amazon S3 with the partitions listed in the AWS Glue Data Catalog for the table. Delete partitions that don't have corresponding data in S3.

Security in AWS Clean Rooms

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Clean Rooms, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations

This documentation helps you understand how to apply the shared responsibility model when using AWS Clean Rooms. It shows you how to configure AWS Clean Rooms to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Clean Rooms resources.

Contents

- Data protection in AWS Clean Rooms
- Using service-linked roles for AWS Clean Rooms
- Data retention in AWS Clean Rooms
- Best practices for data collaborations in AWS Clean Rooms
- Identity and Access Management for AWS Clean Rooms
- Compliance validation for AWS Clean Rooms
- Resilience in AWS Clean Rooms
- Infrastructure security in AWS Clean Rooms
- Access AWS Clean Rooms or AWS Clean Rooms ML using an interface endpoint (AWS PrivateLink)

Data protection in AWS Clean Rooms

The AWS shared responsibility model applies to data protection in AWS Clean Rooms. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see Data Privacy FAQ. For information about data protection in Europe, see the General Data Protection Regulation (GDPR) Center.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with AWS Clean Rooms or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Encryption at rest

AWS Clean Rooms always encrypts all service metadata at rest without requiring any additional configuration. This encryption is automatic when you use AWS Clean Rooms.

Clean Rooms ML encrypts all data stored within the service at rest with AWS KMS. If you choose to provide your own KMS key, the contents of your lookalike models and lookalike segment generation jobs are encrypted at rest with your KMS key.

When using AWS Clean Rooms custom ML models, the service encrypts all data stored at rest with AWS KMS. AWS Clean Rooms supports the use of symmetric customer managed keys that you create, own, and manage to encrypt data at rest. If customer managed keys are not specified, AWS owned keys are used by default.

AWS Clean Rooms uses grants and key policies to access customer managed keys. You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, AWS Clean Rooms won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data. For example, if you attempt to create a trained model from an encrypted ML input channel that AWS Clean Rooms can't access, then the operation would return a `ValidationException` error.

> **Note**
>
> You can use the encryption options in Amazon S3 to protect your data at rest.
> For more information, see Specifying Amazon S3 encryption in the *Amazon S3 User Guide*.

When using an ID mapping table within AWS Clean Rooms, the service encrypts all data stored at rest with AWS KMS. If you choose to provide your own KMS key, the contents of your ID mapping table are encrypted at rest with your KMS key via AWS Entity Resolution. For more details about the required permissions for working with encryptions with an ID mapping workflow, see Create a workflow job role for AWS Entity Resolution in the *AWS Entity Resolution User Guide*.

Encryption in transit

AWS Clean Rooms uses Transport Layer Security (TLS) for encryption in transit. Communication with AWS Clean Rooms is always done over HTTPS so your data is always encrypted in transit, regardless of whether it's stored in Amazon S3, Amazon Athena, or Snowflake. This includes all data in transit when using Clean Rooms ML.

Encrypting underlying data

For more information about how to encrypt your underlying data, see Cryptographic Computing for Clean Rooms.

Key policy

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see Managing access to customer managed keys in the AWS Key Management Service Developer Guide.

To use your customer managed key with your AWS Clean Rooms Custom ML models, the following API operations must be permitted in the key policy:

- `kms:DescribeKey` – Provides the customer managed key details to allow AWS Clean Rooms to validate the key.
- `kms:Decrypt` – Provides access to AWS Clean Rooms to decrypt the encrypted data and use it in related jobs.
- `kms:CreateGrant` - Clean Rooms ML encrypts training and inference images at rest in Amazon ECR by creating grants for Amazon ECR. To learn more, see Encryption at Rest in Amazon ECR. Clean Rooms ML also uses Amazon SageMaker AI to run training and inference jobs, and creates grants for SageMaker AI to encrypt the Amazon EBS volumes attached to the instances as well as the output data in Amazon S3. To learn more, see Protect Data at Rest Using Encryption in Amazon SageMaker AI.
- `kms:GenerateDataKey` - Clean Rooms ML encrypts data at rest stored in Amazon S3 using server-side encryption with AWS KMS keys. To learn more, see Using server-side encryption with AWS KMS keys (SSE-KMS) in Amazon S3.

The following are policy statement examples you can add for AWS Clean Rooms for the following resources:

ML input channel with synthetic data

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
        "Sid": "Allow access to principals authorized to use AWS Clean Rooms ML",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::666666666666:role/ExampleRole"
        },
        "Action": [
            "kms:GenerateDataKey",
            "kms:CreateGrant",
            "kms:Decrypt"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "kms:ViaService": "cleanrooms-ml.us-east-1.amazonaws.com"
            },
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "Encrypt",
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "CreateGrant",
                        "DescribeKey",
                        "RetireGrant",
                        "GenerateDataKey"
                    ]
                },
                "BoolIfExists": {
                    "kms:GrantIsForAWSResource": true
        }
            }
        },
        {
            "Sid": "Allow describe key for principals authorized to use AWS Clean Rooms
 ML",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:role/ExampleRole"
            },
            "Action": [
                "kms:DescribeKey"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "cleanrooms-ml.us-east-1.amazonaws.com"
                }
            }
        },
        {
            "Sid": "Allow grant operations for AWS Clean Rooms ML service principal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "Encrypt",
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "CreateGrant",
                        "DescribeKey",
                        "RetireGrant",
                        "GenerateDataKey"
                    ]
                }
            }
        },
        {
            "Sid": "Allow describe key for AWS Clean Rooms ML service principal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": [
                "kms:DescribeKey"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

ML input channel without synthetic data

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
        "Sid": "Allow access to principals authorized to use AWS Clean Rooms ML",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::666666666666:role/ExampleRole"
        },
        "Action": [
            "kms:DescribeKey",
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "kms:ViaService": "cleanrooms-ml.us-east-1.amazonaws.com"
            }
        }
    },
    {
        "Sid": "Allow access to AWS Clean Rooms ML service principal",
        "Effect": "Allow",
        "Principal": {
            "Service": "cleanrooms-ml.amazonaws.com"
        },
        "Action": [
            "kms:DescribeKey",
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "*"
    }
```

```
  ]
}
```

Trained model job or Trained model inference job

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow grant operations for principals authorized to use AWS
 Clean Rooms ML",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::666666666666:role/ExampleRole"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "cleanrooms-ml.us-east-1.amazonaws.com"
                },
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "Encrypt",
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "CreateGrant",
                        "DescribeKey",
                        "RetireGrant",
                        "GenerateDataKey"
                    ]
                },
                "BoolIfExists": {
                    "kms:GrantIsForAWSResource": true
```

```
                }
            }
        },
        {
            "Sid": "Allow describe key for principals authorized to use AWS Clean
Rooms ML",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:role/ExampleRole"
            },
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "cleanrooms-ml.us-east-1.amazonaws.com"
                }
            }
        },
        {
            "Sid": "Allow grant operations for AWS Clean Rooms ML service
principal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "Encrypt",
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "CreateGrant",
                        "DescribeKey",
                        "RetireGrant",
```

```
                        "GenerateDataKey"
                    ]
                }
            }
        },
        {
            "Sid": "Allow describe key for AWS Clean Rooms ML service principal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms-ml.amazonaws.com"
            },
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

Clean Rooms ML does not support specifying service encryption context or source context in customer managed key policies. Encryption context used by the service internally is visible to customers in CloudTrail.

Encryption for intermediate tables

When using intermediate tables within AWS Clean Rooms, the service encrypts all data stored at rest with AWS Key Management Service (AWS KMS). By default, the service encrypts intermediate table data with a service-managed key. If you choose to provide your own KMS key, the service encrypts your intermediate table at rest with that key. This uses server-side encryption with AWS KMS keys (SSE-KMS).

AWS Clean Rooms uses key policies and service principals to access customer managed keys for intermediate tables. The service encrypts intermediate table data at write time when the table is populated, and decrypts the data transparently when the intermediate table is used in a subsequent analysis.

The KMS key ARN is provided at intermediate table creation time and applies to all versions of the table. You can update the KMS key using the **Edit** action on the intermediate table details page. The new key takes effect on the next populate – each version is encrypted with the key that was specified at the time of population.

> **Note**
>
> Deleting an intermediate table does not require any AWS KMS operations. The encrypted data in Amazon S3 is deleted by the managed storage service.

Required AWS KMS permissions for intermediate tables

To use your customer managed key with intermediate tables, the following API operations must be permitted in the key policy:

- `kms:DescribeKey` – Validates the key state when you create or update an intermediate table.
- `kms:GenerateDataKey` – Encrypts intermediate table data when the table is populated. Amazon S3 calls AWS KMS on behalf of the service to generate data keys for SSE-KMS encryption.
- `kms:Decrypt` – Decrypts intermediate table data when the table is referenced in a subsequent analysis. Amazon S3 calls AWS KMS on behalf of the service to decrypt the data transparently.

Key policy example for intermediate tables

The following is an example key policy for using a customer managed key with intermediate tables:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow Clean Rooms to encrypt and decrypt intermediate table data",
      "Effect": "Allow",
      "Principal": {
        "Service": "cleanrooms.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
```

```
          "kms:EncryptionContext:aws:cleanrooms:intermediateTableArn":
 "arn:aws:cleanrooms:us-east-1:666666666666:membership/membership-id/
intermediatetable/intermediate-table-id"
        },
        "ForAnyValue:ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:cleanrooms:us-east-1:666666666666:membership/member-1-membership-
id",
            "arn:aws:cleanrooms:us-east-1::membership/member-2-membership-id"
          ]
        }
      }
    },
    {
      "Sid": "Allow Clean Rooms to describe key",
      "Effect": "Allow",
      "Principal": {
        "Service": "cleanrooms.amazonaws.com"
      },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow caller to validate key via Clean Rooms",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::666666666666:role/ExampleRole"
      },
      "Action": [
        "kms:DescribeKey",
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "cleanrooms.us-east-1.amazonaws.com",
          "kms:CallerAccount": "666666666666"
        }
      }
    }
  ]
```

```
}
```

The first statement grants the AWS Clean Rooms service principal access for encrypting data during population and decrypting data during analysis. The `kms:EncryptionContext:aws:cleanrooms:intermediateTableArn` condition restricts which intermediate tables can use the key. The `aws:SourceArn` condition provides confused deputy protection by restricting access to specific membership ARNs in your collaboration.

The second statement allows the service to describe the key for validation purposes.

The third statement allows a specific IAM role in the customer's account to perform key validation through AWS Clean Rooms. The permitted actions are DescribeKey, GenerateDataKey, and Decrypt. Access is scoped using `kms:ViaService` and `kms:CallerAccount` conditions. This role is typically the IAM role that you use to interact with AWS Clean Rooms.

Encryption context

AWS Clean Rooms uses the following encryption context when encrypting intermediate table data:

```
{
  "aws:cleanrooms:intermediateTableArn": "arn:aws:cleanrooms:us-
east-1:123456789012:membership/membership-id/intermediatetable/intermediate-table-id"
}
```

Every AWS KMS operation is logged in CloudTrail with the intermediate table ARN, providing a clear audit trail of which resource triggered each encrypt or decrypt call. You can use the `kms:EncryptionContext` condition key in your key policy to restrict which intermediate tables can use the key.

Using service-linked roles for AWS Clean Rooms

AWS Clean Rooms uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS Clean Rooms. Service-linked roles are predefined by AWS Clean Rooms and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS Clean Rooms easier because you don’t have to manually add the necessary permissions. AWS Clean Rooms defines the permissions of its service-

linked roles, and unless defined otherwise, only AWS Clean Rooms can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete a service-linked role only after first deleting their related resources. This protects your AWS Clean Rooms resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for AWS Clean Rooms

AWS Clean Rooms uses the service-linked role named **AWSServiceRoleForAWSCleanRooms** – to publish Clean Rooms-related CloudWatch metrics to your AWS account.

The AWSServiceRoleForAWSCleanRooms service-linked role trusts the following services to assume the role:

- `cleanrooms.amazonaws.com`

The role permissions policy named AWSCleanRoomsServiceRolePolicy allows AWS Clean Rooms to complete the following actions on the specified resources:

- Action: `cloudwatch:PutMetricData` on `all AWS resources, restricted to the AWS Clean Rooms namespace`

You must configure permissions to allow your users, groups, or roles to create, edit, or delete a service-linked role. For more information, see Service-linked role permissions in the *IAM User Guide*.

Creating a service-linked role for AWS Clean Rooms

You can use the IAM console to create a service-linked role with the **AWSServiceRoleForAWSCleanRooms** use case. In the AWS CLI or the AWS API, create a service-linked role with the `cleanrooms.amazonaws.com` service name. For more information, see Creating a service-linked role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a service-linked role for AWS Clean Rooms

AWS Clean Rooms does not allow you to edit the AWSServiceRoleForAWSCleanRooms service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for AWS Clean Rooms

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

To delete the **AWSServiceRoleForAWSCleanRooms**, you must first delete all collaborations and memberships in your AWS account.

> **Note**
>
> If the AWS Clean Rooms service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForAWSCleanRooms service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for AWS Clean Rooms service-linked roles

AWS Clean Rooms supports using service-linked roles in all of the Regions where the service is available. For more information, see AWS Regions and endpoints.

Data retention in AWS Clean Rooms

Any data that is temporarily read into an AWS Clean Rooms collaboration is deleted after the query completes.

When you create a lookalike model, Clean Rooms ML reads your training data, transforms it into a format suitable for our ML model, and stores the trained model parameters inside Clean Rooms ML. Clean Rooms ML does not retain a copy of your training data. AWS Clean Rooms SQL queries do not retain any of your data after the query has run. Clean Rooms ML then uses the trained model to summarize the behavior of all of your users. Clean Rooms ML stores a user-level data set for each user in your data for as long as your lookalike model is active.

When you start a lookalike segment generation job, Clean Rooms ML reads the seed data, reads the behavior summaries from the associated lookalike model, and creates a lookalike segment that is stored within the AWS Clean Rooms service. Clean Rooms ML does not retain a copy of your seed data. Clean Rooms ML stores the user-level output of the job as long as the job is active.

If your seed data comes from an SQL query, the output of that query is only stored in the service for the duration of the job. The results of the query are encrypted at rest and in transit.

If you want to remove your lookalike model or lookalike segment generation job data, use the API to delete it. Clean Rooms ML asynchronously deletes all data associated with the model or job. Once this process is complete, Clean Rooms ML deletes the metadata for the model or job and it is no longer visible in the API. Clean Rooms ML retains deleted data for 3 days for disaster recovery prevention. Once the job or model is no longer visible in the API and 3 days have passed, all data associated with the model or job has been permanently deleted.

Best practices for data collaborations in AWS Clean Rooms

This topic describes the best practices for conducting data collaborations in AWS Clean Rooms.

AWS Clean Rooms follows the AWS Shared Responsibility Model. AWS Clean Rooms offers analysis rules that you can configure to strengthen your ability to protect sensitive data in a collaboration. The analysis rules that you configure in AWS Clean Rooms will enforce the restrictions (query controls and query output controls) that you have configured. You are responsible for determining the restrictions and configuring analysis rules accordingly.

Data collaborations might involve more than just your use of AWS Clean Rooms. To help you maximize the benefit of data collaborations, we recommend that you perform the following best practices with your use of AWS Clean Rooms and specifically with analysis rules.

Topics

- Best practices with AWS Clean Rooms

- Best practices for using analysis rules in AWS Clean Rooms

Best practices with AWS Clean Rooms

You're responsible for assessing the risk of each data collaboration and comparing it to your privacy requirements such as external and internal compliance programs and policies. We recommend that you take additional actions with your use of AWS Clean Rooms. These actions might help further manage risks and help guard against third-party attempts to re-identify your data (for example, differencing attacks or side-channel attacks).

For example, consider conducting due diligence on your other collaborators and enter into legal agreements with them *before* engaging in a collaboration. To monitor the use of your data, also consider adopting other audit mechanisms with your use of AWS Clean Rooms.

Best practices for using analysis rules in AWS Clean Rooms

Analysis rules in AWS Clean Rooms allow you to restrict the queries that can be run by setting query controls on a configured table. For example, you can set a query control for how a configured table can be joined and which columns can be selected. You can also restrict the query output through setting query result controls such as aggregation thresholds on output rows. The service rejects any query and removes rows that don’t comply with the analysis rules set by members on their configured tables in the query.

We recommend the following *10 best practices* for using analysis rules on your configured table:

- Create separate configured tables for separate query use cases (for example, audience planning or attribution). You can create multiple configured tables with the same underlying AWS Glue table.
- Specify columns in the analysis rule (for example, dimension columns, list columns, join columns) that are necessary for queries in a collaboration. This might help mitigate the risk of differencing attacks or enabling other members to reverse engineer your data. Use the **allowlist columns** feature to note other columns that you might want to make queryable in the future. To customize the columns that can be used for a certain collaboration, create additional configured tables with the same underlying AWS Glue table.
- Specify the functions in the analysis rule that are necessary for analysis in the collaboration. This can help mitigate risk from rare function errors that can present information on an individual data point. To customize the functions that can be used for a certain collaboration, create additional configured tables with the same underlying AWS Glue table.

- Add aggregation constraints on any columns whose values at a row-level are sensitive. This includes columns in your configured table that also exist in other collaboration members' tables and analysis rules as an aggregation constraint. This also includes columns in your configured table that aren't queryable, that is, columns that are in your configured table but are not in the analysis rule. Aggregation constraints can help mitigate risk from correlating query results with data outside the collaboration.
- Create test collaborations and analysis rules to test restrictions created with specified analysis rules.
- Review collaborator configured tables and members' analysis rules on configured tables to check that they match what was agreed upon for the collaboration. This can help mitigate risk from other members engineering their own data to run queries that weren't agreed upon.
- Review the example query provided (console only) that is enabled on your configured table after you set up the analysis rule.

> **Note**
>
> In addition to the provided example query, other queries are possible based on the analysis rule and other collaboration member tables and analysis rules.

- You can add or update an analysis rule for a configured table in a collaboration. When you do, review all the collaborations where the configured table is associated and its resulting impact. This helps make sure that no collaborations use obsolete analysis rules.
- Review the queries run in the collaboration to check that the queries match the use cases or queries that were agreed upon for the collaboration. (The queries are available in the query logs when the **Query logging** feature is turned on.) This can help mitigate risk from members running analysis that was not agreed upon and potential attacks such as side channel attacks.
- Review the configured table columns used in collaboration members' analysis rules and in queries to check that they match what was agreed upon in the collaboration. (The queries are available in the query logs when that feature is turned on.) This can help mitigate risk from other members engineering their own data to do queries that weren't agreed upon.

Identity and Access Management for AWS Clean Rooms

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in)

and *authorized* (have permissions) to use AWS Clean Rooms resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How AWS Clean Rooms works with IAM
- Identity-based policy examples for AWS Clean Rooms
- AWS managed policies for AWS Clean Rooms
- Troubleshooting AWS Clean Rooms identity and access
- Cross-service confused deputy prevention
- IAM behaviors for AWS Clean Rooms ML
- IAM behaviors for Clean Rooms ML Custom Models

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshooting AWS Clean Rooms identity and access)
- **Service administrator** - determine user access and submit permission requests (see How AWS Clean Rooms works with IAM)
- **IAM administrator** - write policies to manage access (see Identity-based policy examples for AWS Clean Rooms)

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users or your company's single sign-on authentication are examples of federated identities. When you sign in as a federated identity, your

administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signature Version 4 signing process in the *AWS General Reference*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center User Guide* and Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see AWS account root user credentials and IAM identities in the *AWS General Reference*.

Federated identity

As a best practice, require human users to use federation with an identity provider to access AWS services using temporary credentials.

A *federated identity* is a user from your enterprise directory, web identity provider, or Directory Service that accesses AWS services using credentials from an identity source. Federated identities assume roles that provide temporary credentials.

For centralized access management, we recommend AWS IAM Identity Center. For more information, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. By default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A

user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An

explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.

- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS Clean Rooms works with IAM

Before you use IAM to manage access to AWS Clean Rooms, learn what IAM features are available to use with AWS Clean Rooms.

IAM features you can use with AWS Clean Rooms

IAM feature	AWS Clean Rooms support
Identity-based policies	Yes
Resource-based policies	Partial
Policy actions	Yes
Policy resources	Yes

IAM feature	AWS Clean Rooms support
Policy condition keys (service-specific)	Partial
ACLs	No
ABAC (tags in policies)	Yes
Temporary credentials	Yes
Forward access sessions (FAS)	Yes
Service roles	Yes
Service-linked roles	No

To get a high-level view of how AWS Clean Rooms and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for AWS Clean Rooms

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for AWS Clean Rooms

To view examples of AWS Clean Rooms identity-based policies, see Identity-based policy examples for AWS Clean Rooms.

Resource-based policies within AWS Clean Rooms

Supports resource-based policies: Partial

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

The AWS Clean Rooms service supports only one type of resource-based policy called a *configured lookalike model managed resource policy*, which is attached to a configured lookalike model. This policy defines which principals can perform actions on the configured lookalike model.

To learn how to attach a resource-based policy to a configured lookalike model, see **IAM behaviors for AWS Clean Rooms ML**.

Policy actions for AWS Clean Rooms

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of AWS Clean Rooms actions, see Actions defined by AWS Clean Rooms in the *Service Authorization Reference*.

Policy actions in AWS Clean Rooms use the following prefix before the action.

```
cleanrooms
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
```

```
        "cleanrooms:action1",
        "cleanrooms:action2"
           ]
```

To view examples of AWS Clean Rooms identity-based policies, see Identity-based policy examples for AWS Clean Rooms.

Policy resources for AWS Clean Rooms

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of AWS Clean Rooms resource types and their ARNs, see Resources defined by AWS Clean Rooms in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by AWS Clean Rooms.

To view examples of AWS Clean Rooms identity-based policies, see Identity-based policy examples for AWS Clean Rooms.

Policy condition keys for AWS Clean Rooms

Supports service-specific policy condition keys: Partial

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To learn how AWS Clean Rooms ML uses policy condition keys, see **IAM behaviors for AWS Clean Rooms ML**.

ACLs in AWS Clean Rooms

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with AWS Clean Rooms

Supports ABAC (tags in policies): Yes

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with AWS Clean Rooms

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Forward access sessions for AWS Clean Rooms

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for AWS Clean Rooms

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break AWS Clean Rooms functionality. Edit service roles only when AWS Clean Rooms provides guidance to do so.

Service-linked roles for AWS Clean Rooms

Supports service-linked roles: No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for AWS Clean Rooms

By default, users and roles don't have permission to create or modify AWS Clean Rooms resources. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by AWS Clean Rooms, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for AWS Clean Rooms in the *Service Authorization Reference*.

Topics

- Policy best practices
- Using the AWS Clean Rooms console
- Allow users to view their own permissions

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Clean Rooms resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies

adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.

- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the AWS Clean Rooms console

To access the AWS Clean Rooms console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Clean Rooms resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the AWS Clean Rooms console, also attach the AWS Clean Rooms `FullAccess` or `ReadOnly` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS managed policies for AWS Clean Rooms

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS managed policy: `AWSCleanRoomsReadOnlyAccess`

You can attach `AWSCleanRoomsReadOnlyAccess` to your IAM principals.

This policy grants read-only permissions to resources and metadata in an `AWSCleanRoomsReadOnlyAccess` collaboration.

Permissions details

This policy includes the following permissions:

- `CleanRoomsRead` – Allows principals read-only access to the service.
- `ConsoleDisplayTables` – Allows principals read-only access to the AWS Glue metadata needed to show data about the underlying AWS Glue tables on the console.
- `ConsoleLogSummaryQueryLogs` – Allows principals to see the query logs.
- `ConsoleLogSummaryObtainLogs` – Allows principals to retrieve the log results.

For a JSON listing of the policy details, see AWSCleanRoomsReadOnlyAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: `AWSCleanRoomsFullAccess`

You can attach `AWSCleanRoomsFullAccess` to your IAM principals.

This policy grants administrative permissions that allow full access (read, write, and update) to resources and metadata in an AWS Clean Rooms collaboration. This policy includes access to perform queries.

Permissions details

This policy includes the following permissions:

- `CleanRoomsAccess` – Grants full access to all actions on all resources for AWS Clean Rooms.
- `PassServiceRole` – Grants access to pass a service role to only the service (`PassedToService` condition) that has "cleanrooms" in its name.
- `ListRolesToPickServiceRole` – Allows principals to list all their roles in order to choose a service role when using AWS Clean Rooms.
- `GetRoleAndListRolePoliciesToInspectServiceRole` – Allows principals to see the service role and corresponding policy in IAM.
- `ListPoliciesToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `GetPolicyToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `ConsoleDisplayTables` – Allows principals read-only access to the AWS Glue metadata needed to show data about the underlying AWS Glue tables on the console.
- `ConsolePickQueryResultsBucketListAll` – Allows principals to choose an Amazon S3 bucket from a list of all available S3 buckets into which their query results are written.
- `SetQueryResultsBucket` – Allows principals to choose an S3 bucket into which their query results are written.
- `ConsoleDisplayQueryResults` – Allows principals to show the query results to the customer, read from the S3 bucket.
- `WriteQueryResults` – Allows principals to write the query results into a customer-owned S3 bucket.
- `EstablishLogDeliveries` – Allows principals to deliver query logs to a customer's Amazon CloudWatch Logs log group.
- `SetupLogGroupsDescribe` – Allows principals to use the Amazon CloudWatch Logs log group creation process.
- `SetupLogGroupsCreate` – Allows principals to create an Amazon CloudWatch Logs log group.
- `SetupLogGroupsResourcePolicy` – Allows principals to set up a resource policy on the Amazon CloudWatch Logs log group.
- `ConsoleLogSummaryQueryLogs` – Allows principals to see the query logs.
- `ConsoleLogSummaryObtainLogs` – Allows principals to retrieve the log results.

For a JSON listing of the policy details, see AWSCleanRoomsFullAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: `AWSCleanRoomsFullAccessNoQuerying`

You can attach `AWSCleanRoomsFullAccessNoQuerying` to your IAM principals.

This policy grants administrative permissions that allow full access (read, write, and update) to resources and metadata in an AWS Clean Rooms collaboration. This policy excludes access to perform queries.

Permissions details

This policy includes the following permissions:

- `CleanRoomsAccess` – Grants full access to all actions on all resources for AWS Clean Rooms, except for querying in collaborations.
- `CleanRoomsNoQuerying` – Explicitly denies `StartProtectedQuery` and `UpdateProtectedQuery` to prevent querying.
- `PassServiceRole` – Grants access to pass a service role to only the service (`PassedToService` condition) that has "cleanrooms" in its name.
- `ListRolesToPickServiceRole` – Allows principals to list all their roles in order to choose a service role when using AWS Clean Rooms.
- `GetRoleAndListRolePoliciesToInspectServiceRole` – Allows principals to see the service role and corresponding policy in IAM.
- `ListPoliciesToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `GetPolicyToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `ConsoleDisplayTables` – Allows principals read-only access to the AWS Glue metadata needed to show data about the underlying AWS Glue tables on the console.
- `EstablishLogDeliveries` – Allows principals to deliver query logs to a customer's Amazon CloudWatch Logs log group.
- `SetupLogGroupsDescribe` – Allows principals to use the Amazon CloudWatch Logs log group creation process.
- `SetupLogGroupsCreate` – Allows principals to create an Amazon CloudWatch Logs log group.
- `SetupLogGroupsResourcePolicy` – Allows principals to set up a resource policy on the Amazon CloudWatch Logs log group.
- `ConsoleLogSummaryQueryLogs` – Allows principals to see the query logs.

- `ConsoleLogSummaryObtainLogs` – Allows principals to retrieve the log results.
- `cleanrooms` – Manage collaborations, analysis templates, configured tables, memberships, and associated resources within the AWS Clean Rooms service. Perform various operations such as creating, updating, deleting, listing, and retrieving information about these resources.
- `iam` – Pass service roles with names containing "`cleanrooms`" to the AWS Clean Rooms service. List roles, policies, and inspect service roles and policies related to the AWS Clean Rooms service.
- `glue` – Retrieve information about databases, tables, partitions, and schemas from AWS Glue. This is required for the AWS Clean Rooms service to display and interact with the underlying data sources.
- `logs` – Manage log deliveries, log groups, and resource policies for CloudWatch Logs. Query and retrieve logs related to the AWS Clean Rooms service. These permissions are necessary for monitoring, auditing, and troubleshooting purposes within the service.

The policy also explicitly denies the actions `cleanrooms:StartProtectedQuery` and `cleanrooms:UpdateProtectedQuery` to prevent users from directly executing or updating protected queries, which should be done through the AWS Clean Rooms controlled mechanisms.

For a JSON listing of the policy details, see AWSCleanRoomsFullAccessNoQuerying in the *AWS Managed Policy Reference Guide*.

AWS managed policy: `AWSCleanRoomsMLReadOnlyAccess`

You can attach `AWSCleanRoomsMLReadOnlyAccess` to your IAM principals.

This policy grants read-only permissions to resources and metadata in an `AWSCleanRoomsMLReadOnlyAccess` collaboration.

This policy includes the following permissions:

- `CleanRoomsConsoleNavigation` – Grants access to view the screens of the AWS Clean Rooms console.
- `CleanRoomsMLRead` – Allows principals read-only access to the Clean Rooms ML service.
- `PassCleanRoomsResources` – Grants access to pass specified AWS Clean Rooms resources.

For a JSON listing of the policy details, see AWSCleanRoomsMLReadOnlyAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: `AWSCleanRoomsMLFullAccess`

You can attach `AWSCleanRoomsMLFullAcces` to your IAM principals. This policy grants administrative permissions that allow full access (read, write, and update) to resources and metadata needed by Clean Rooms ML.

Permissions details

This policy includes the following permissions:

- `CleanRoomsMLFullAccess` – Grants access to all Clean Rooms ML actions.
- `PassServiceRole` – Grants access to pass a service role to only the service (`PassedToService` condition) that has "cleanrooms-ml" in its name.
- `CleanRoomsConsoleNavigation` – Grants access to view the screens of the AWS Clean Rooms console.
- `CollaborationMembershipCheck` – When you start an audience generation (lookalike segment) job within a collaboration, the Clean Rooms ML service calls `ListMembers` to check that the collaboration is valid, the caller is an active member, and the configured audience model owner is an active member. This permission is always required; the console navigation SID is only required for console users.
- `PassCleanRoomsResources` – Grants access to pass specified AWS Clean Rooms resources.
- `AssociateModels` – Allows principals to associate a Clean Rooms ML model with your collaboration.
- `TagAssociations` – Allows principals to add tags to the association between a lookalike model and a collaboration.
- `ListRolesToPickServiceRole` – Allows principals to list all their roles in order to choose a service role when using AWS Clean Rooms.
- `GetRoleAndListRolePoliciesToInspectServiceRole` – Allows principals to see the service role and corresponding policy in IAM.
- `ListPoliciesToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `GetPolicyToInspectServiceRolePolicy` – Allows principals to see the service role and corresponding policy in IAM.
- `ConsoleDisplayTables` – Allows principals read-only access to the AWS Glue metadata needed to show data about the underlying AWS Glue tables on the console.

- `ConsolePickOutputBucket` – Allows principals to select Amazon S3 buckets for configured audience model outputs.
- `ConsolePickS3Location` – Allows principals to select the location within a bucket for configured audience model outputs.
- `ConsoleDescribeECRRepositories` – Allows principals to describe Amazon ECR repositories and images.

For a JSON listing of the policy details, see AWSCleanRoomsMLFullAccess in the *AWS Managed Policy Reference Guide*.

AWS Clean Rooms updates to AWS managed policies

View details about updates to AWS managed policies for AWS Clean Rooms since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the AWS Clean Rooms Document history page.

Change	Description	Date
AWSCleanRoomsFullAccessNoQuerying– Update to existing policy	Added cleanrooms:UpdateConfiguredTableAllowedColumns and cleanrooms:UpdateConfiguredTableReference to CleanRoomsAccess.	July 29, 2025
AWSCleanRoomsMLReadOnlyAccess – Update to existing policy	Added PassCleanRoomsResources to AWSCleanRoomsMLReadOnlyAccess. Added PassCleanRoomsResources and ConsoleDescribeECRRepositories to AWSCleanRoomsMLFullAccess.	January 10, 2025
AWSCleanRoomsFullAccessNoQuerying – Update to existing policy	Added cleanrooms:BatchGetSchemaAnalysisRule to CleanRoomsAccess.	May 13, 2024
AWSCleanRoomsFullAccess – Update to existing policy	Updated the Statement ID in AWSCleanRoomsFullAccess from ConsolePickQueryResultsBucket to SetQueryResultsBucket in this policy to better represent the permissions	March 21, 2024

Change	Description	Date
	since the permissions are needed for setting the query results bucket both with and without the console.	
AWSCleanRoomsMLReadOnlyAccess – New policy AWSCleanRoomsMLFullAccess – New policy	Added AWSCleanRoomsMLReadOnlyAccess and AWSCleanRoomsMLFullAccess to support AWS Clean Rooms ML.	November 29, 2023
AWSCleanRoomsFullAccessNoQuerying – Update to existing policy	Added cleanrooms:CreateAnalysisTemplate, cleanrooms:GetAnalysisTemplate, cleanrooms:UpdateAnalysisTemplate, cleanrooms:DeleteAnalysisTemplate, cleanrooms:ListAnalysisTemplates, cleanrooms:GetCollaborationAnalysisTemplate, cleanrooms:BatchGetCollaborationAnalysisTemplate, and cleanrooms:ListCollaborationAnalysisTemplates to CleanRoomsAccess to enable the new analysis templates feature.	July 31, 2023
AWSCleanRoomsFullAccessNoQuerying – Update to existing policy	Added cleanrooms:ListTagsForResource, cleanrooms:UntagResource, and cleanrooms:TagResource to CleanRoomsAccess to enable resource tagging.	March 21, 2023
AWS Clean Rooms started tracking changes	AWS Clean Rooms started tracking changes for its AWS managed policies.	January 12, 2023

Troubleshooting AWS Clean Rooms identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS Clean Rooms and IAM.

Topics

- I am not authorized to perform an action in AWS Clean Rooms
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my AWS Clean Rooms resources

I am not authorized to perform an action in AWS Clean Rooms

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but does not have the fictional `cleanrooms:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cleanrooms:GetWidget on resource: my-example-widget
```

In this case, Mateo's policy must be updated to allow him to access the *`my-example-widget`* resource using the `cleanrooms:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to AWS Clean Rooms.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS Clean Rooms. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my AWS Clean Rooms resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role.

To learn more, consult the following:

- To learn whether AWS Clean Rooms supports these features, see How AWS Clean Rooms works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Cross-service confused deputy prevention

The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In AWS, cross-service impersonation can result in the confused deputy problem. Cross-service impersonation can occur when one service (the *calling service*) calls another service (the *called service*). The calling service can be manipulated to use its permissions to act on another customer's resources in a way it should not otherwise have permission to access. To prevent this, AWS provides tools that help you protect your data for all services with service principals that have been given access to resources in your account.

We recommend using the `aws:SourceArn` global condition context keys in resource policies to limit the permissions that AWS Clean Rooms gives another service to the resource. Use `aws:SourceArn` if you want only one resource to be associated with the cross-service access.

The most effective way to protect against the confused deputy problem is to use the `aws:SourceArn` global condition context key with the full ARN of the resource. In AWS Clean Rooms, you also have to compare against the `sts:ExternalId` condition key.

The value of `aws:SourceArn` must be set to the ARN of the membership of the assumed role.

The following example shows how you can use the `aws:SourceArn` global condition context key in AWS Clean Rooms to prevent the confused deputy problem.

> **Note**
>
> The example policy applies to the trust policy of the service role that AWS Clean Rooms uses to access data and metadata for a configured table.
> The value for *`<query-runner-membership-id>`* needs to be set to the membership ID of the query runner.
> All members of the collaboration can view the configured table metadata so each membership ARN must be included in the list of membership ARNs.

> **Note**
>
> When a service role is created through the AWS Clean Rooms console, all current members of the collaboration are included in the confused deputy condition by default.
> If you are adding new members to a collaboration that already has configured tables associated to it, make sure that you update your service role's confused deputy condition with the new member's membership ARN.
> If you do not update your service role's confused deputy condition after adding a new member, that new member will not be able to access information in AWS Clean Rooms that is retrieved using that role.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIfExternalIdMatches",
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringLike": {
                    "sts:ExternalId": "arn:aws:*:us-east-1:*:dbuser:*/<query-
runner-membership-id>*"
                }
            }
        },
        {
            "Sid": "AllowIfSourceArnMatches",
            "Effect": "Allow",
            "Principal": {
                "Service": "cleanrooms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ForAnyValue:ArnEquals": {
                    "aws:SourceArn": [
                        "arn:aws:cleanrooms:us-
east-1:111122223333:membership/<member-1-membership-id>",
                        "arn:aws:cleanrooms:us-
east-1:444455556666:membership/<member-2-membership-id>"
                    ]
                }
            }
        }
    ]
}
```

IAM behaviors for AWS Clean Rooms ML

Cross-account jobs

Clean Rooms ML allows certain resources created by one AWS account to be securely accessed in their account by another AWS account. When a client in AWS account A calls `StartAudienceGenerationJob` on a `ConfiguredAudienceModel` resource owned by AWS account B, Clean Rooms ML creates two ARNs for the job. One ARN in AWS account A and another in AWS account B. The ARNs are identical except for their AWS account.

Clean Rooms ML creates two ARNs for the job to ensure that both accounts can apply their own IAM policies to the jobs. For example, both accounts can use tag-based access control and apply policies from their AWS organization. The job processes data from both accounts, so both accounts can delete the job and its associated data. Neither account can block the other account from deleting the job.

There is only one job execution and both accounts can see the job when they call `ListAudienceGenerationJobs`. Both accounts can call the `Get`, `Delete`, and `Export` APIs on the job using the ARN with their own AWS account ID.

Neither AWS account can access the job when using an ARN with the other AWS account ID.

The name of the job must be unique within an AWS account. The name in AWS account B is *`$accountA-$name`*. The name chosen by AWS account A is prefixed with AWS account A when the job is viewed in AWS account B.

In order for a cross-account `StartAudienceGenerationJob` to succeed, AWS account B must allow that action on both the new job in AWS account B and the `ConfiguredAudienceModel` in AWS account B using a resource policy similar to the following example:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Clean-Rooms-CAMA-ID",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "111122223333"
                ]
            },
            "Action": [
                "cleanrooms-ml:StartAudienceGenerationJob"
            ],
            "Resource": [
                "arn:aws:cleanrooms-ml:us-east-1:444455556666:configured-
audience-model/id",
                "arn:aws:cleanrooms-ml:us-east-1:444455556666:audience-
generation-job/*"
```

```
            ],
            "Condition":{"StringEquals":{"cleanrooms-ml:CollaborationId":"UUID"}}
        }
    ]
}
```

Note

This AWS Clean Rooms ML resource policy references two different AWS account IDs to support cross-account audience generation:

- **111122223333** - This is the account that contains the principal (user, role, or service) authorized to start audience generation jobs. This account initiates the ML processing workflow.
- **444455556666** - This is the account that owns the AWS Clean Rooms ML resources (the configured audience model and audience generation jobs). This account hosts the ML models and manages the job execution.

Additional Configuration Notes:

- **Statement ID (Sid)**: Replace `CAMA-ID` with your actual AWS Clean Rooms Audience Model Application (CAMA) identifier to make the policy statement easily identifiable.
- **Resource IDs**: Replace *`id`* with the actual ID of your configured audience model, and *`UUID`* with your specific collaboration ID.
- **Condition**: The `cleanrooms-ml:CollaborationId` condition ensures that audience generation jobs can only be started within the context of the specified AWS Clean Rooms collaboration, providing an additional security boundary.

This cross-account configuration enables scenarios where one organization manages the ML models and infrastructure while allowing authorized partners to initiate audience generation processes within the bounds of their collaboration agreement.

If you use the AWS Clean Rooms ML API to create a configured lookalike model with `manageResourcePolicies` set to true, AWS Clean Rooms creates this policy for you.

Additionally, the identity policy of the caller in AWS account A needs `StartAudienceGenerationJob` permission on `arn:aws:cleanrooms-ml:us-west-1:AccountA:audience-generation-job/*`. So there are three IAM Resources for Action `StartAudienceGenerationJob`: the AWS account A job, the AWS account B job, and the AWS account B `ConfiguredAudienceModel`.

> **Warning**
>
> The AWS account that started the job receives an AWS CloudTrail audit log event about the job. The AWS account that owns the `ConfiguredAudienceModel` does not receive a AWS CloudTrail audit log event.

Tagging jobs

When you set the `childResourceTagOnCreatePolicy=FROM_PARENT_RESOURCE` parameter of `CreateConfiguredAudienceModel`, all lookalike segment generation jobs within your account that are created from that configured lookalike model default to having the same tags as the configured lookalike model. The configured lookalike model is the parent and the lookalike segment generation job is the child.

If you are creating a job within your own account, the request tags of the job override the parent tags. Jobs created by other accounts never create tags in your account. If you set `childResourceTagOnCreatePolicy=FROM_PARENT_RESOURCE` and another account creates a job, there are two copies of the job. The copy in your account has the parent resource tags and the copy in the job submitter's account has tags from the request.

Validating collaborators

When granting permissions to other members of an AWS Clean Rooms collaboration, the resource policy should include the condition key `cleanrooms-ml:CollaborationId`. This enforces that the `collaborationId` parameter is included in the StartAudienceGenerationJob request. When the `collaborationId` parameter is included in the request, Clean Rooms ML validates that the collaboration exists, the job submitter is an active member of the collaboration, and the configured lookalike model owner is an active member of the collaboration.

When AWS Clean Rooms manages your configured lookalike model resource policy (the `manageResourcePolicies` parameter is TRUE in CreateConfiguredAudienceModelAssociation

request), this condition key will be set in the resource policy. Therefore, you must specify the `collaborationId` in StartAudienceGenerationJob.

Cross-account access

Only `StartAudienceGenerationJob` can be called across accounts. All other Clean Rooms ML APIs can only be used with resources in your own account. This ensures that your training data, lookalike model configuration, and other information stays private.

Clean Rooms ML never reveals Amazon S3 or AWS Glue locations across accounts. The training data location, configured lookalike model output location, and lookalike segment generation job seed location are never visible across accounts. Unless query logging is enabled in the collaboration, whether the seed data comes from an SQL query and the query itself are not visible across accounts. If you `Get` an audience generation job that another account submitted, the service does not show the seed location.

IAM behaviors for Clean Rooms ML Custom Models

Cross-account jobs

Clean Rooms ML allows certain resources associated with a collaboration created by one AWS account to be securely accessed in their account by another AWS account. A client in AWS account A with the member ability to run queries can call `CreateTrainedModel`, `CreateMLInputChannel`, or `StartTrainedModelInferenceJob` on a `ConfiguredModelAlgorithmAssociation` resource owned by another member in the collaboration, provided the `ConfiguredModelAlgorithmAssociation` is allowed by the custom analysis rule created with `CreateConfiguredTableAnalysisRule`.

Additionally, any active member of a collaboration can delete data associated with a trained model or ML input channel via the `DeleteTrainedModelOutput` and `DeleteMLInputChannelData` APIs.

Cross-account access

Clean Rooms ML allows users to retrieve metadata about resources created by other accounts via the `GetCollaboration` and `ListCollaboration` APIs. Clean Rooms ML does not reveal KMS key ARNs, tags, environment variables, or hyperparameters (for the `TrainedModel` action) to other accounts.

Membership and collaboration access

When accessing membership and collaboration resources in context of Clean Rooms ML custom models, a user's identity policy needs permissions to the actions `cleanrooms:PassMembership`, `cleanrooms:PassCollaboration`, or both. All APIs that accept `membershipId` need the `cleanrooms:PassMembership` permission, and all APIs that accept `collaborationId` need the `cleanrooms:PassCollaboration` permission. A sample identity policy for a role that can call `createTrainedModel` in the context of a membership ID that can call `GetCollaborationTrainedModel` in the context of a collaboration ID are provided.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCleanroomsMLActions",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:PassCollaboration",
                "cleanrooms:PassMembership"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "AllowMembershipAccess",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:GetMembership"
            ],
            "Resource": [
                "arn:aws:cleanrooms:us-east-1:111122223333:membership/memberId"
            ]
        },
        {
            "Sid": "AllowCollaborationAccess",
            "Effect": "Allow",
            "Action": [
                "cleanrooms:GetCollaboration"
```

```
            ],
            "Resource": [
                "arn:aws:cleanrooms:us-
east-1:111122223333:collaboration/collaborationId"
            ]
        }
    ]
}
```

Compliance validation for AWS Clean Rooms

Our new AWS sign-up experience is not designed for regulated workloads. If you're using our new AWS sign-up experience, but you want to use AWS for regulated workloads, you can sign up for AWS (advanced) or activate advanced features for your AWS environment.

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. For more information about your compliance responsibility when using AWS services, see AWS Security Documentation.

Resilience in AWS Clean Rooms

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS Clean Rooms

As a managed service, AWS Clean Rooms is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access AWS Clean Rooms through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Network security

When AWS Clean Rooms reads from your S3 bucket during query execution, the traffic between AWS Clean Rooms and Amazon S3 is securely routed through the AWS private network. In-flight traffic is signed using Amazon Signature Version 4 protocol (SIGv4) and encrypted using HTTPS. This traffic is authorized based on the IAM service role which you have set up for your configured table.

You can connect programmatically to AWS Clean Rooms through an endpoint. For a list of service endpoints, see AWS Clean Rooms endpoints and quotas in the *AWS General Reference*.

All service endpoints are HTTPS-only. You can use Amazon Virtual Private Cloud (VPC) endpoints in case you want to connect to AWS Clean Rooms from your VPC and do not want to have internet connectivity. For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

You can assign IAM policies to your IAM principals which make use of the aws:SourceVpce context keys to restrict your IAM principal to only be able to make calls to AWS Clean Rooms through a VPC endpoint and not over the internet.

Access AWS Clean Rooms or AWS Clean Rooms ML using an interface endpoint (AWS PrivateLink)

You can use AWS PrivateLink to create a private connection between your virtual private cloud (VPC) and AWS Clean Rooms or AWS Clean Rooms ML. You can access AWS Clean Rooms or AWS Clean Rooms ML as if it were in your VPC, without the use of an internet gateway, NAT device, VPN connection, or Direct Connect connection. Instances in your VPC don't need public IP addresses to access AWS Clean Rooms.

You establish this private connection by creating an *interface endpoint*, powered by AWS PrivateLink. We create an endpoint network interface in each subnet that you enable for the interface endpoint. These are requester-managed network interfaces that serve as the entry point for traffic destined for AWS Clean Rooms.

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

Considerations for AWS Clean Rooms

Before you set up an interface endpoint for AWS Clean Rooms, review Considerations in the *AWS PrivateLink Guide*.

AWS Clean Rooms and AWS Clean Rooms ML support making calls to all of their API actions through the interface endpoint.

VPC endpoint policies are not supported for AWS Clean Rooms or AWS Clean Rooms ML. By default, full access to AWS Clean Rooms and AWS Clean Rooms ML is allowed through the interface endpoint. Alternatively, you can associate a security group with the endpoint network interfaces to control traffic to AWS Clean Rooms or AWS Clean Rooms ML through the interface endpoint.

Create an interface endpoint for AWS Clean Rooms

You can create an interface endpoint for AWS Clean Rooms or AWS Clean Rooms ML using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Create an interface endpoint in the *AWS PrivateLink Guide*.

Create an interface endpoint for AWS Clean Rooms using the following service name.

```
com.amazonaws.region.cleanrooms
```

Create an interface endpoint for AWS Clean Rooms ML using the following service name.

```
com.amazonaws.region.cleanrooms-ml
```

If you enable private DNS for the interface endpoint, you can make API requests to AWS Clean Rooms using its default Regional DNS name. For example, `cleanrooms-ml.us-east-1.amazonaws.com`.

Monitoring AWS Clean Rooms

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Clean Rooms and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Clean Rooms, report when something is wrong, and take automatic actions when appropriate:

- *Amazon CloudWatch Logs* enables you to monitor, store, and access your log files from Amazon EC2 instances, AWS CloudTrail, and other sources. Amazon CloudWatch Logs can monitor information in the log files and notify you when certain thresholds are met. You can also archive your log data in highly durable storage. For more information, see the Amazon CloudWatch Logs User Guide.

  Clean Rooms ML allows cross-account jobs for certain API actions. The AWS account that started the job receives the AWS CloudTrail audit log event for the job. For more information, see IAM behaviors for AWS Clean Rooms ML
- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.
- *Amazon EventBridge* is a serverless event bus service that makes it easy to connect your applications with data from a variety of sources. EventBridge delivers a stream of real-time data from your own applications, Software-as-a-Service (SaaS) applications, and AWS services and routes that data to targets such as Lambda. This enables you to monitor events that happen in services, and build event-driven architectures. For more information, see the Amazon EventBridge User Guide and the Amazon EventBridge Events Reference.
- *AWS Billing and Cost Management* is a service that provides features to help you pay your bills and optimize your costs. Amazon Web Services bills your account based on usage, ensuring that you pay only for what you use. AWS also enables you to apply cost allocation tags to your resources to track and categorize your AWS costs. For example, in AWS Clean Rooms, you can apply tags to collaboration memberships to track costs per collaboration. For more information, see the AWS Billing User Guide.

Analysis logging in AWS Clean Rooms

Analysis logging is a feature in AWS Clean Rooms. When you create a collaboration and turn on **Analysis logging**, members can store relevant logs from queries or jobs in Amazon CloudWatch Logs.

With analysis logs, members can determine if the queries comply with the analysis rules and align with the collaboration agreement. In addition, analysis logs help support audits.

When logs are exported, AWS Clean Rooms records an event in the same log group for every log export. You receive the log export event if you ran the analysis, or if you contributed a table that the analysis referenced. The event shows who requested the export, which analysis the export was requested for, the destination in Amazon S3, and whether the request succeeded. For more information about exporting logs, see Exporting query analysis logs.

When **Analysis logging** is turned on, AWS Clean Rooms generates logs at two stages: when the analysis runs and when the analysis completes.

The following table describes the fields included in analysis logs.

Field	Delivered when	Description
`eventId`	Always delivered	The unique identifier for the analysis run. For queries, this is the same as the `protectedQueryID` . For jobs, this is the same as the `protectedJobID` .
`eventTimestamp`	Always delivered	The time the analysis ran.
`collaborationId`	Always delivered	The unique identifier for the collaboration.
`logStage`	Always delivered	Indicates whether the log is from submission validation (SUBMISSION) or terminal completion (TERMINATION).

Field	Delivered when	Description
`analysisCategory`	Always delivered	The type of analysis: SQL (for queries) or PYSPARK (for jobs).
`analysisCategoryVersion`	Always delivered	The version of the analysis category (for example, 1.0).
`schemaName`	An analysis runs	The name of the configured table association referenced in the analysis.
`configuredTableId`	An analysis runs	The unique identifier for the configured table referenced in the analysis.
`analysisTemplateArn`	An analysis runs	The analysis template that was run (appears depending on analysis rule).
`parameters`	An analysis runs	The parameter values (appears depending on the analysis text).
`queryText`	A query runs	The SQL definition of the query that was run. If there are parameters, they are labeled as `:parametervalue` .
`directQueryAnalysisRuleType`	A query runs	The type of analysis rule for the configured table.
`directQueryAnalysisRulePolicy`	A query runs	The analysis rule policy including allowed analyses and allowed analysis providers.
`queryValidationErrors`	A query runs	The query errors at query validation.
`resultReceivers`	A query runs	The members designated to receive query results.
`queryRunners`	A query runs	The members designated to run queries.

Field	Delivered when	Description
`resultRegions`	A query runs	The AWS Regions where results can be delivered.
`analysisTemplateArtifactHashList`	A job runs	The hash list of artifacts associated with the analysis template.
`directJobAnalysisRuleType`	A job runs	The type of analysis rule for the configured table.
`directJobAnalysisRulePolicy`	A job runs	The analysis rule policy including allowed analyses and allowed analysis providers.
`jobValidationErrors`	A job runs	The job errors at job validation.
`jobRunners`	A job runs	The members designated to run jobs.
`status`	An analysis finishes	The execution status of the analysis.
`errorCode`	An analysis finishes	The error code when an analysis failed to execute properly.
`errorMessage`	An analysis finishes	The error message when an analysis failed to execute properly.
`memberSchemaMapping`	Always delivered	A mapping of member accounts to their schema information.
`memberDisplayNames`	Always delivered	A mapping of member accounts to their display names.

Field	Delivered when	Description
additionalAnalyses	Synthetic data is created	The ARN of the additional analyses resource permitted for the synthetic-data analysis. Delivered to members whose configured table is referenced in the query, not to the member who created the ML input channel.
isSynthetic	Synthetic data is created	Whether the analysis created privacy-enhanced synthetic data.
epsilon	Synthetic data is created	The epsilon value required to successfully create privacy-enhanced synthetic data.
maxMembershipInferenceAttackScore	Synthetic data is created	The maximum membership inference attack score allowed to successfully create privacy-enhanced synthetic data.
machineLearningInputChannelArn	Synthetic data is created	The ARN of the machine learning input channel used to generate the synthetic data. Delivered to the member who created the ML input channel and to members whose configured tables are referenced.
analysisId	Logs are exported	The unique identifier of the analysis for which logs were exported.
callerAccountId	Logs are exported	The account ID of the member who requested the export.
s3OutputPath	Logs are exported	The Amazon S3 destination where the exported logs were written.
operationName	Logs are exported	The operation that generated the export event.
logExportValidationErrors	Logs are exported	Any validation errors during the log export.

Example analysis logs

The following examples show representative analysis log entries. The examples use fictional identifiers.

Example Query submission log


```
{
  "eventId": "ffffffff-0000-4111-8222-333333333333",
  "eventTimestamp": 1723392245000,
  "collaborationId": "11111111-2222-4333-8444-555555555555",
  "logStage": "SUBMISSION",
  "analysisCategory": "SQL",
  "analysisCategoryVersion": "1.0",
  "schemaName": "publisher-conversions",
  "configuredTableId": "99999999-aaaa-4bbb-8ccc-dddddddddddd",
  "analysisTemplateArn": "arn:aws:cleanrooms:us-
east-1:444455556666:membership/66666666-7777-4888-8999-aaaaaaaaaaaa/
analysistemplate/77777777-8888-4999-8aaa-bbbbbbbbbbbb",
  "parameters": "{\"minimum_overlap\":\"100\"}",
  "queryText": "SELECT COUNT(*) FROM publisher_conversions WHERE overlap
 >= :minimum_overlap",
  "directQueryAnalysisRuleType": "AGGREGATION",
  "directQueryAnalysisRulePolicy": "{\"v1\":{\"custom\":{\"allowedAnalyses\":
[\"arn:aws:cleanrooms:us-east-1:444455556666:membership/66666666-7777-4888-8999-
aaaaaaaaaaaa/analysistemplate/77777777-8888-4999-8aaa-bbbbbbbbbbbb\"]}}}",
  "queryValidationErrors": null,
  "resultReceivers": "[\"444455556666\"]",
  "queryRunners": "[\"444455556666\"]",
  "resultRegions": "[\"us-east-1\"]",
  "memberSchemaMapping": "{\"444455556666\":\"publisher-conversions\"}",
  "memberDisplayNames": "{\"444455556666\":\"Example Publisher\"}"
}
```


Example Query completion log


```
{
  "eventId": "ffffffff-0000-4111-8222-333333333333",
  "eventTimestamp": 1723392305000,
  "collaborationId": "11111111-2222-4333-8444-555555555555",
  "logStage": "TERMINATION",
  "analysisCategory": "SQL",
  "analysisCategoryVersion": "1.0",
```

```
  "status": "COMPLETED",
  "errorCode": null,
  "errorMessage": null
}
```

Example Job submission log

```
{
  "eventId": "00000000-1111-4222-8333-444444444444",
  "eventTimestamp": 1723392365000,
  "collaborationId": "11111111-2222-4333-8444-555555555555",
  "logStage": "SUBMISSION",
  "analysisCategory": "PYSPARK",
  "analysisCategoryVersion": "1.0",
  "schemaName": "publisher-conversions",
  "configuredTableId": "99999999-aaaa-4bbb-8ccc-dddddddddddd",
  "analysisTemplateArn": "arn:aws:cleanrooms:us-
east-1:444455556666:membership/66666666-7777-4888-8999-aaaaaaaaaaaa/
analysistemplate/77777777-8888-4999-8aaa-bbbbbbbbbbbb",
  "analysisTemplateArtifactHashList": "[\"abc123def456\"]",
  "directJobAnalysisRuleType": "CUSTOM",
  "directJobAnalysisRulePolicy": "{\"v1\":{\"custom\":{\"allowedAnalyses\":
[\"arn:aws:cleanrooms:us-east-1:444455556666:membership/66666666-7777-4888-8999-
aaaaaaaaaaaa/analysistemplate/77777777-8888-4999-8aaa-bbbbbbbbbbbb\"]}}}",
  "jobValidationErrors": null,
  "jobRunners": "[\"444455556666\"]",
  "memberSchemaMapping": "{\"444455556666\":\"publisher-conversions\"}",
  "memberDisplayNames": "{\"444455556666\":\"Example Publisher\"}"
}
```

Example Job completion log

```
{
  "eventId": "00000000-1111-4222-8333-444444444444",
  "eventTimestamp": 1723392605000,
  "collaborationId": "11111111-2222-4333-8444-555555555555",
  "logStage": "TERMINATION",
  "analysisCategory": "PYSPARK",
  "analysisCategoryVersion": "1.0",
  "status": "COMPLETED",
  "errorCode": null,
  "errorMessage": null
```

```
}
```

Delivery, recipients, and log groups

You don't need to perform any actions outside of AWS Clean Rooms to set up query logs and job logs. AWS Clean Rooms creates log groups for collaborations after each collaboration member creates a membership.

Members who can query, members who can run queries and jobs, members who can receive results, and members whose configured tables are referenced in the query receive an analysis log.

The member who can query and member who can receive results receive query logs for each configured table that is referenced in the query. If they don't own the configured table, they can't view the configured table ID (`configuredTableId`).

The member who can run queries and jobs and member who can receive results receive job logs for each configured table that is referenced in the job. If they don't own the configured table, they can't view the configured table ID (`configuredTableId`).

If a member has multiple configured table associations referenced in the analysis, they receive an analysis log for each configured table.

Members with tables used to create an intermediate table don't receive analysis logs when analyses are run on the intermediate table.

AWS Clean Rooms creates logs for queries that contain unsupported and supported SQL. For more details, see the AWS Clean Rooms SQL Reference.

AWS Clean Rooms also creates logs when queries or jobs reference configured tables that are not associated with the collaboration.

Logs might contain information about incorrect SQL.

Query and job logs indicate the status of a query but don't report whether query output was delivered. They confirm that a query or job was submitted by the member who can query. Query logs also confirm that the query contains supported SQL in AWS Clean Rooms and references configured tables that are associated with the collaboration.

> **Note**
>
> AWS Clean Rooms doesn't produce a log if the query was canceled after AWS Clean Rooms validated its compliance with analysis rules and during query processing.

If you delete the log group, you must re-create the log group manually with the same log group name (collaboration ID of the collaboration). Or, you can turn the logging off and on in your membership.

For more information about how to turn on analysis logging, see Creating a collaboration.

For more information about Amazon CloudWatch Logs, see the Amazon CloudWatch Logs User Guide.

Recommended actions for query and job logs

We recommend that members periodically take the following actions:

- To verify that the queries and jobs match the use cases or queries that were agreed upon for the collaboration, review the queries and jobs that are run in the collaboration.

  For more information about how to view recent queries, see Viewing recent queries.

  For more information about how to view recent jobs, see Viewing recent jobs.
- To verify that the configured table columns match what was agreed upon for the collaboration, review the configured table columns that are used in collaboration members' analysis rules and in queries.

  For more information about how to view the configured columns, see Viewing tables and analysis rules.

Detailed monitoring with CloudWatch in AWS Clean Rooms

Detailed monitoring is an opt-in feature in AWS Clean Rooms. When you create a collaboration and turn on **Detailed monitoring**, members who can run queries and configured payors can choose to receive detailed observability metrics in their CloudWatch account.

With detailed monitoring, members can monitor query performance, track resource utilization, and create dashboards for operational insights. These metrics help with capacity planning, cost analysis, and troubleshooting performance issues.

> **Note**
>
> Metric availability varies based on configured table data sources:
>
> - For analyses that reference configured tables with Athena as the data source, the `BytesRead` and `QueryFileScanned` metrics always return 0.
> - For analyses that reference configured tables with Snowflake as the data source, the `BytesRead`, `QueryFileScanned`, and `RecordsRead` metrics always return 0.
> - For analyses that reference multiple configured tables with different data sources (e.g., Snowflake and S3-backed AWS Glue tables), `BytesRead` and `QueryFileScanned` return metrics only for S3-backed AWS Glue tables. `RecordsRead` returns metrics for S3-backed AWS Glue tables and Athena tables.

Types of metrics

AWS Clean Rooms publishes two categories of metrics to CloudWatch:

Aggregated detailed monitoring metrics

These metrics provide aggregate-level visibility across analyses:

`AnalysisRuntime`

Shows the total duration of an analysis in milliseconds.

`CRPUConsumed`

The CRPU (Clean Room Processing Units) consumed for a particular analysis. For more information about CRPU calculation, see AWS Clean Rooms Pricing.

`ConcurrentVCPUs`

Concurrent vCPU utilized at a given point in time (related to service quotas).

`AnalysesCount`

Number of analyses run over a period of time for a specified dimension.

`ConcurrentQueries`

Concurrent query count at a given point in time (related to service quotas).

Query-level detailed monitoring metrics

These metrics provide detailed insights into individual query execution:

`BytesRead`

Total data (in bytes) read from source tables for an analysis.

`BytesWritten`

Total data (in bytes) written for an analysis.

`RecordsRead`

Total number of records read from source tables for an analysis.

`RecordsWritten`

Total number of records returned in the analysis result.

`QueryFileScanned`

Total number of data files scanned for an analysis.

`Memory utilization`

Memory resource used for an analysis.

`vCPU utilization`

vCPU resource used for an analysis.

`Disk utilization`

Storage resource used for an analysis.

Metric dimensions

Metrics are organized by the following dimensions:

- **Analysis Status**: Completed
- **Analysis Type**: SparkSQL
- **Membership**: The membership ID
- **Query**: Individual query ID (for query-level metrics)
- **Account/Region**: Your AWS account and Region

All metrics are published to the `AWS/CleanRooms` namespace in CloudWatch.

Who can access metrics

Detailed monitoring metrics are available to:

- **Query runners**: Members who can run queries in the collaboration
- **Configured payors**: Members designated to pay for query execution costs

Members who only contribute data cannot access detailed monitoring metrics.

Pricing

AWS Clean Rooms does not charge for publishing metrics to CloudWatch. However, standard CloudWatch charges apply for metric storage, dashboard usage, and alarms. For more information, see Amazon CloudWatch Pricing.

Logging AWS Clean Rooms API calls using AWS CloudTrail

AWS Clean Rooms is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Clean Rooms. CloudTrail captures all API calls for AWS Clean Rooms as events. The calls captured include calls from the AWS Clean Rooms console and code calls to the AWS Clean Rooms API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Clean Rooms. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AWS Clean Rooms, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Clean Rooms information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Clean Rooms, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing events with CloudTrail Event history.

For an ongoing record of events in your AWS account, including events for AWS Clean Rooms, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple Regions
- Receiving CloudTrail log files from multiple accounts

All AWS Clean Rooms actions are logged by CloudTrail and are documented in the *AWS Clean Rooms API Reference*.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root user or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity element.

Understanding AWS Clean Rooms log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of

the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

> **Note**
>
> AWS Clean Rooms redacts only certain request parameters that its API defines. If you include additional fields that aren't part of the API, CloudTrail records them exactly as you sent them and doesn't redact them. Don't include passwords or other sensitive information in fields that aren't part of the API.

Example AWS Clean Rooms CloudTrail events

The following examples demonstrate CloudTrail events for:

Topics

- StartProtectedQuery (successful)
- StartProtectedQuery (failed)

StartProtectedQuery (successful)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID",
        "arn": "arn:aws:sts::123456789012:assumed-role/query-runner/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/query-runner",
                "accountId": "123456789012",
                "userName": "query-runner"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-04-07T19:34:32Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-04-07T19:53:32Z",
    "eventSource": "cleanrooms.amazonaws.com",
    "eventName": "StartProtectedQuery",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "203.0.113.1",
    "userAgent": "aws-internal/3",
    "requestParameters": {
        "resultConfiguration": {
            "outputConfiguration": {
                "s3": {
                    "resultFormat": "CSV",
                    "bucket": "cleanrooms-queryresults-jdoe-test",
                    "keyPrefix": "test"
                }
            }
        },
        "sqlParameters": "***",
        "membershipIdentifier": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "type": "SQL"
    },
    "responseElements": {
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "protectedQuery": {
            "createTime": 1680897212.279,
            "id": "f5988bf1-771a-4141-82a8-26fcc4e41c9f",
            "membershipArn": "arn:aws:cleanrooms:us-east-2:123456789012:membership/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "membershipId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "resultConfiguration": {
                "outputConfiguration": {
                    "s3": {
                        "bucket": "cleanrooms-queryresults-jdoe-test",
                        "keyPrefix": "test",
                        "resultFormat": "CSV"
                    }
                }
            },
            "sqlParameters": "***",
            "status": "SUBMITTED"
```

```
        }
    },
    "requestID": "7464211b-2277-4b55-9723-fb4f259aefd2",
    "eventID": "f7610f5e-74b9-420f-ae43-206571ebcbf7",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

StartProtectedQuery (failed)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID",
        "arn": "arn:aws:sts::123456789012:assumed-role/query-runner/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/query-runner",
                "accountId": "123456789012",
                "userName": "query-runner"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-04-07T19:34:32Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-04-07T19:47:27Z",
    "eventSource": "cleanrooms.amazonaws.com",
    "eventName": "StartProtectedQuery",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "203.0.113.1",
    "userAgent": "aws-internal/3",
    "errorCode": "ValidationException",
```

```
    "requestParameters": {
        "resultConfiguration": {
            "outputConfiguration": {
                "s3": {
                    "resultFormat": "CSV",
                    "bucket": "cleanrooms-queryresults-jdoe-test",
                    "keyPrefix": "test"
                }
            }
        },
        "sqlParameters": "***",
        "membershipIdentifier": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "type": "SQL"
    },
    "responseElements": {
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "message": "Column(s) [identifier] is not allowed in select"
    },
    "requestID": "e29f9f74-8299-4a83-9d18-5ddce7302f07",
    "eventID": "c8ee3498-8e4e-44b5-87e4-ab9477e56eb5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```


Integrating AWS Clean Rooms into event-driven applications using Amazon EventBridge

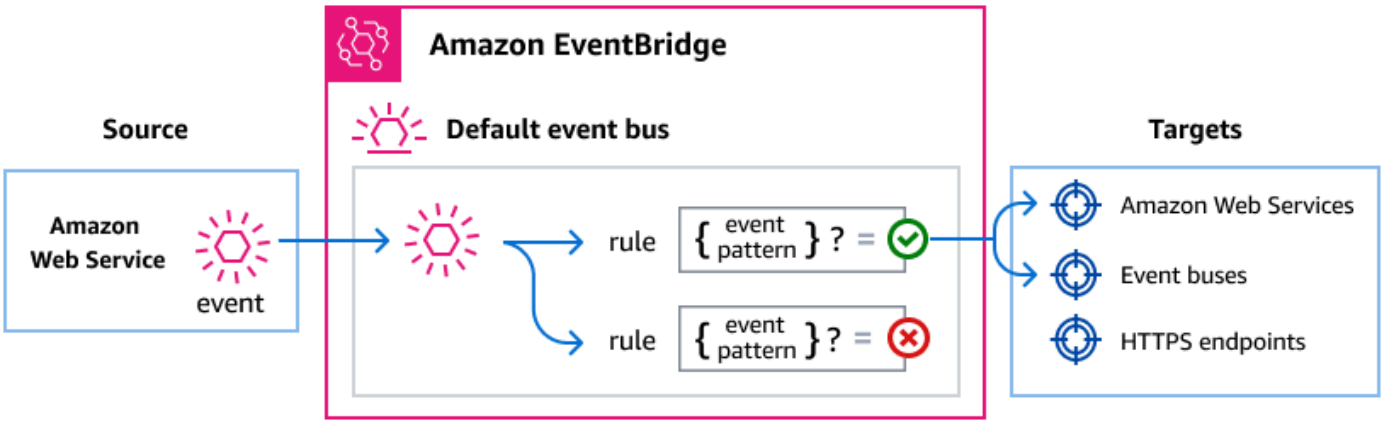
You can incorporate AWS Clean Rooms into event-driven applications (EDAs) that use events that occur in AWS Clean Rooms to communicate between application components and initiate downstream processes. You do this by using Amazon EventBridge to route events from AWS Clean Rooms to other software components. Amazon EventBridge is a serverless service that uses events to connect application components together, making it easier for you to integrate AWS services like AWS Clean Rooms into event-driven architectures without additional code and operations.

Event-driven architecture is a style of building loosely-coupled software systems that work together by emitting and responding to events. In this model, an events represents a change in a resource or environment.

Here's how EventBridge works with AWS Clean Rooms:

As with many AWS services, AWS Clean Rooms generates and sends events to the EventBridge default *event bus*. An event bus is a router that receives events and routes them to the destinations, or *targets*, that you specify. Targets can include other AWS services, custom applications, and SaaS partner applications.

EventBridge routes events according to *rules* you create on the event bus. For each rule, you specify a filter, or *event pattern*, to select only the events you want. Whenever an event is sent to the event bus, EventBridge compares it against each rule. If the event matches the rule, EventBridge routes the event to the specified target(s).



For example, suppose you want to know every time a new AWS Clean Rooms collaboration is created in your account. You could create a rule on the default event bus. In the rule you would create an event pattern that specified events from AWS Clean Rooms named **`Collaboration Created`**. Every time EventBridge received an event matching those properties, it would route the event to the specified workflow.

AWS Clean Rooms events

AWS services can send events directly to the EventBridge default event bus. In addition, AWS CloudTrail sends events originating from numerous AWS services to EventBridge as well. These events can include API calls, console sign ins and actions, service events, and CloudTrail Insights. For more information, see AWS service events delivered via AWS CloudTrail in the *EventBridge User Guide*.

For a full list of AWS Clean Rooms events sent to EventBridge, refer to the AWS Clean Rooms topic in the *EventBridge Events Reference*.

Event detail type	Description
Analysis Template Created	The **Analysis Template** owner and all active members of the collaboration are notified when an **Analysis Template** is **Created**.
Analysis Template Updated	The **Analysis Template** owner and all active members of the collaboration that have visibility into the update are notified when an **Analysis Template** is **Updated**.
Analysis Template Deleted	The **Analysis Template** owner and all active members of the collaboration are notified when an **Analysis Template** is **Deleted**.
Collaboration Created	The **Collaboration** owner is notified when a **Collaboration** is **Created**.
Collaboration Updated	The **Collaboration** owner and all active members of the collaboration that have visibility into the update are notified when a **Collaboration** is **Updated**.
Collaboration Change Request Created	The **Collaboration** owner and all active members of the collaboration are notified when a **Collaboration Change Request** is **Created**.
Collaboration Change Request Approved	The **Collaboration** owner and all active members of the collaboration are notified when a **Collaboration Change Request** is **Approved**.
Collaboration Change Request Cancelled	The **Collaboration** owner and all active members of the collaboration are notified when a **Collaboration Change Request** is **Cancelled**.
Collaboration Change Request Committed	The **Collaboration** owner and all active members of the collaboration are notified when a **Collaboration Change Request** is **Committed**.

Event detail type	Description
Configured Table Association Created	The **Configured Table Association** owner and all active members of the collaboration are notified when a **Configured Table Association** is **Created**.
Configured Table Association Updated	The **Configured Table Association** owner and all active members of the collaboration that have visibility into the update are notified when a **Configured Table Association** is **Updated**.
Configured Table Association Deleted	The **Configured Table Association** owner and all active members of the collaboration are notified when a **Configured Table Association** is **Deleted**.
Configured Table Association Analysis Rule Created	The **Configured Table Association Analysis Rule** owner and all active members of the collaboration are notified when a **Configured Table Association Analysis Rule** is **Created**.
Configured Table Association Analysis Rule Updated	The **Configured Table Association Analysis Rule** owner and all active members of the collaboration that have visibility into the update are notified when a **Configured Table Association Analysis Rule** is **Updated**.
Configured Table Association Analysis Rule Deleted	The **Configured Table Association Analysis Rule** owner and all active members of the collaboration are notified when a **Configured Table Association Analysis Rule** is **Deleted**.
Id Mapping Table Created	The **Id Mapping Table** owner and all active members of the collaboration are notified when an **Id Mapping Table** is **Created**.
Id Mapping Table Updated	The **Id Mapping Table** owner and all active members of the collaboration that have visibility into the update are notified when an **Id Mapping Table** is **Updated**.
Id Mapping Table Deleted	The **Id Mapping Table** owner and all active members of the collaboration are notified when an **Id Mapping Table** is **Deleted**.

Event detail type	Description
Id Namespace Association Created	The **Id Namespace Association** owner and all active members of the collaboration are notified when an **Id Namespace Association** is **Created**.
Id Namespace Association Updated	The **Id Namespace Association** owner and all active members of the collaboration that have visibility into the update are notified when an **Id Namespace Association** is **Updated**.
Id Namespace Association Deleted	The **Id Namespace Association** owner and all active members of the collaboration are notified when an **Id Namespace Association** is **Deleted**.
Invited To Collaboration	The invited **member** is notified when they are invited to a **Collaboration**.
Membership Created	The **Membership** owner and all active members of the collaboration are notified when a **Membership** is **Created**.
Membership Updated	The **Membership** owner is notified when a **Membership** is **Updated**, unless the **membership** was removed from the collaboration in which case all active members of the collabora tion are notified.
Membership Deleted	The **Membership** owner and all active members of the collaboration are notified when a **Membership** is **Deleted**.
Protected Job Submitted	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** is **Submitted**.
Protected Job Started	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** is **Started**.
Protected Job Cancelling	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** is **Cancelling**.

Event detail type	Description
Protected Job Cancelled	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** is **Cancelled**.
Protected Job Succeeded	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** has **Succeeded**.
Protected Job Failed	The **Job Runner**, **Job Payer** and **Results Receiver** of a **Protected Job** are notified when the **Protected Job** has **Failed**.
Protected Query Submitted	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** is **Submitted**.
Protected Query Started	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** is **Started**.
Protected Query Cancelling	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** is **Cancelling**.
Protected Query Cancelled	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** is **Cancelled**.
Protected Query Succeeded	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** has **Succeeded**.
Protected Query Failed	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** has **Failed**.

Event detail type	Description
Protected Query Timed Out	The **Query Runner**, **Query Payer** and **Results Receiver(s)** of a **Protected Query** are notified when the **Protected Query** has **Timed Out**.

Routing AWS Clean Rooms events using EventBridge

To have EventBridge route AWS Clean Rooms events to a target, you must create a rule. Each rule contains an event pattern, which EventBridge matches against each event received on the event bus. If the event data matches the specified event pattern, EventBridge routes that event to the rule's target(s).

For comprehensive instructions on creating event bus rules, see Creating rules that react to events in the *EventBridge User Guide*.

Creating event patterns that match AWS Clean Rooms events

Each event pattern is a JSON object that contains:

- (Optional): A `source` attribute that identifies the service sending the event. For AWS Clean Rooms events, the source is `aws.cleanrooms`.
- (Optional): A `detail-type` attribute that contains an array of the event names to match.
- (Optional): A `detail` attribute containing any other event data on which to match.

For example, the following event pattern matches against all Id Namespace Association Updated events where the collaboration was deleted from AWS Clean Rooms:

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Namespace Association Updated"],
  "detail": {
    "status": ["COLLABORATION_DELETED"]
  }
}
```

For more information on writing event patterns, see Event patterns in the *EventBridge User Guide*.

AWS Clean Rooms events detail reference

All events from AWS services have a common set of fields containing metadata about the event, such as the AWS service that is the source of the event, the time the event was generated, the account and region in which the event took place, and others. For definitions of these general fields, see Event structure in the *Amazon EventBridge Events Reference*.

In addition, each event has a `detail` field that contains data specific to that particular event. The reference below defines the detail fields for the various AWS Clean Rooms events.

When using EventBridge to select and manage AWS Clean Rooms events, it's useful to keep the following in mind:

- The `source` field for all events from AWS Clean Rooms is set to `aws.cleanrooms`.
- The `detail-type` field specifies the event type.

  For example, `Collaboration Created`.
- The `detail` field contains the data that is specific to that particular event.

For information on constructing event patterns that enable rules to match AWS Clean Rooms events, see Event patterns in the *Amazon EventBridge User Guide*.

Analysis Template Created event

Below are the detail fields for the `Analysis Template Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Analysis Template Created"]
}
```

Analysis Template Updated event

Below are the detail fields for the `Analysis Template Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Analysis Template Updated"]
}
```

Analysis Template Deleted event

Below are the detail fields for the `Analysis Template Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Analysis Template Deleted"]
}
```

Collaboration Created event

Below are the detail fields for the `Collaboration Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Created"]
}
```

Collaboration Updated event

Below are the detail fields for the `Collaboration Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Updated"]
}
```

Collaboration Change Request Created event

Below are the detail fields for the `Collaboration Change Request Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Change Request Created"]
}
```

Collaboration Change Request Approved event

Below are the detail fields for the `Collaboration Change Request Approved` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Change Request Approved"]
}
```

Collaboration Change Request Cancelled event

Below are the detail fields for the `Collaboration Change Request Cancelled` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Change Request Cancelled"]
}
```

Collaboration Change Request Committed event

Below are the detail fields for the `Collaboration Change Request Committed` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Collaboration Change Request Committed"]
}
```

Configured Table Association Created event

Below are the detail fields for the `Configured Table Association Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Created"]
}
```

Configured Table Association Updated event

Below are the detail fields for the `Configured Table Association Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Updated"]
```

```
}
```

Configured Table Association Deleted event

Below are the detail fields for the `Configured Table Association Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Deleted"]
}
```

Configured Table Association Analysis Rule Created event

Below are the detail fields for the `Configured Table Association Analysis Rule Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Analysis Rule Created"]
}
```

Configured Table Association Analysis Rule Updated event

Below are the detail fields for the `Configured Table Association Analysis Rule Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Analysis Rule Updated"]
}
```

Configured Table Association Analysis Rule Deleted event

Below are the detail fields for the `Configured Table Association Analysis Rule Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Configured Table Association Analysis Rule Deleted"]
}
```

Id Mapping Table Created event

Below are the detail fields for the `Id Mapping Table Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Mapping Table Created"]
}
```

Id Mapping Table Updated event

Below are the detail fields for the `Id Mapping Table Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Mapping Table Updated"]
}
```

Id Mapping Table Deleted event

Below are the detail fields for the `Id Mapping Table Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Mapping Table Deleted"]
}
```

Id Namespace Association Created event

Below are the detail fields for the `Id Namespace Association Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Namespace Association Created"]
}
```

Id Namespace Association Updated event

Below are the detail fields for the `Id Namespace Association Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Namespace Association Updated"]
}
```

> **Note**
>
> When a collaboration is deleted, the event is captured in the Id Namespace Association Updated event in the `detail.status` field.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Namespace Association Updated"],
  "detail": {
    "status": ["COLLABORATION_DELETED"]
}
```

Id Namespace Association Deleted event

Below are the detail fields for the `Id Namespace Association Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Id Namespace Association Deleted"]
}
```

Invited To Collaboration event

Below are the detail fields for the `Invited To Collaboration` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Invited To Collaboration"]
}
```

Membership Created event

Below are the detail fields for the `Membership Created` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Membership Created"]
}
```

Membership Updated event

Below are the detail fields for the `Membership Updated` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Membership Updated"]
}
```

Membership Deleted event

Below are the detail fields for the `Membership Deleted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Membership Deleted"]
}
```

Protected Job Submitted event

Below are the detail fields for the `Protected Job Submitted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Submitted"]
}
```

Protected Job Started event

Below are the detail fields for the `Protected Job Started` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Started"]
}
```

Protected Job Cancelling event

Below are the detail fields for the `Protected Job Cancelling` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Cancelling"]
}
```

Protected Job Cancelled event

Below are the detail fields for the `Protected Job Cancelled` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Cancelled"]
}
```

Protected Job Succeeded event

Below are the detail fields for the `Protected Job Succeeded` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Succeeded"]
}
```

Protected Job Failed event

Below are the detail fields for the `Protected Job Failed` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Job Failed"]
}
```

Protected Query Submitted event

Below are the detail fields for the `Protected Query Submitted` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Submitted"]
}
```

Protected Query Started event

Below are the detail fields for the `Protected Query Started` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Started"]
}
```

Protected Query Cancelling event

Below are the detail fields for the `Protected Query Cancelling` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Cancelling"]
}
```

Protected Query Cancelled event

Below are the detail fields for the `Protected Query Cancelled` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Cancelled"]
}
```

Protected Query Succeeded event

Below are the detail fields for the `Protected Query Succeeded` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Succeeded"]
}
```

Protected Query Failed event

Below are the detail fields for the `Protected Query Failed` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Failed"]
}
```

Protected Query Timed Out event

Below are the detail fields for the `Protected Query Timed Out` event.

```
{
  "source": ["aws.cleanrooms"],
  "detail-type": ["Protected Query Timed Out"]
}
```

Cost Allocation Tagging

AWS Clean Rooms supports using Cost Allocation Tags to track your AWS costs. You activate these tags on the AWS Billing and Cost Management dashboard. AWS uses the tags to categorize your costs and deliver a monthly cost allocation report to you. User-defined cost allocation tags can be applied to AWS Clean Rooms resources to help track and allocate costs across your collaboration.

Taggable Resources for Cost Allocation

The following table lists the AWS Clean Rooms resources that incur charges and can be tagged with user-defined cost allocation tags for tracking and organizing costs.

Billed Resource	Tagged resource
SQL Query	Membership
PySpark Job	Membership
ML Training Job	Membership
ML Inference Job	Membership

Billed Resource	Tagged resource
Synthetic Data Generation	Membership
Lookalike Model Training	Lookalike Model
Lookalike Segment Export	Lookalike Segment

Creating AWS Clean Rooms resources with AWS CloudFormation

AWS Clean Rooms is integrated with AWS CloudFormation, a service that helps you to model and set up your AWS resources. As a result of this integration, you can spend less time creating and managing your resources and infrastructure. You create a template that describes all the AWS resources that you want, and CloudFormation provisions and configures those resources for you. Examples of resources include collaborations, configured tables, configured table associations, and memberships.

When you use CloudFormation, you can reuse your template to set up your AWS Clean Rooms resources consistently and repeatedly. Describe your resources once, and then provision the same resources over and over in multiple AWS accounts and AWS Regions.

AWS Clean Rooms and CloudFormation templates

To provision and configure resources for AWS Clean Rooms and related services, you must understand CloudFormation templates. Templates are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your CloudFormation stacks. If you're unfamiliar with JSON or YAML, you can use CloudFormation Designer to help you get started with CloudFormation templates. For more information, see What is CloudFormation Designer? in the *AWS CloudFormation User Guide*.

AWS Clean Rooms supports creating collaborations, configured tables, configured table associations, and memberships in CloudFormation. For more information, including examples of JSON and YAML templates for collaborations, configured tables, configured table associations, and memberships, see the AWS Clean Rooms and AWS Clean Rooms ML resource type references in the *AWS CloudFormation User Guide*.

Learn more about CloudFormation

To learn more about CloudFormation, see the following resources:

- AWS CloudFormation
- AWS CloudFormation User Guide
- CloudFormation API Reference

- AWS CloudFormation Command Line Interface User Guide

Quotas for AWS Clean Rooms

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is specific to an AWS Region. You can request increases for some quotas, and other quotas can't be increased.

To view the quotas for AWS Clean Rooms, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **AWS Clean Rooms**.

To request a quota increase, see Requesting a Quota Increase in the *Service Quotas User Guide*. If the quota isn't yet available in Service Quotas, use the Service limit increase form.

Topics

- AWS Clean Rooms quotas
- AWS Clean Rooms ML quotas

AWS Clean Rooms quotas

Your AWS account has the following quotas related to AWS Clean Rooms.

Name	Default	Adjustable	Description
Analysis rule size	Each supported Region: 100 Kilobytes	No	Maximum size of JSON for an analysis rule
Analysis templates per membership	Each supported Region: 100	Yes	Maximum number of analysis templates per membership
CR.1X workers per PySpark job	Each supported Region: 128	Yes	Maximum number of CR.1X workers per PySpark job

Name	Default	Adjusted	Description
CR.4X workers per PySpark job	Each supported Region: 32	Yes	Maximum number of CR.4X workers per PySpark job
Collaborations created per account	Each supported Region: 500	Yes	Maximum number of collaborations created per account
Columns per configured table allowlist	Each supported Region: 100	Yes	Maximum number of columns that can be allowlisted per configured table
Concurrent PySpark job vCPU usage per account	us-east-1: 1,024 Each of the other supported Regions: 512	Yes	Maximum total vCPU usage of all concurrently running PySpark jobs per account
Concurrent PySpark jobs per account	us-east-1: 5 Each of the other supported Regions: 2	Yes	Maximum number of concurrently running PySpark jobs per account
Concurrent SQL queries per account	us-east-1: 5 Each of the other supported Regions: 2	Yes	Maximum number of concurrently running SQL queries per account
Concurrent SQL query vCPU usage per account	Each supported Region: 512	Yes	Maximum total vCPU usage of all concurrently running SQL queries per account

Name	Default	Adjusted	Description
Concurrent ongoing job per membership	Each supported Region: 1	No	Maximum number of concurrent ongoing jobs per membership
Concurrent ongoing log exports per account	Each supported Region: 10	Yes	Maximum number of concurrent ongoing log exports per account
Concurrent ongoing log exports per membership	Each supported Region: 5	Yes	Maximum number of concurrent ongoing log exports per membership
Concurrent ongoing queries per membership	Each supported Region: 5	No	Maximum number of concurrent ongoing queries per membership
Configured audience model associations per membership	Each supported Region: 5	Yes	Maximum number of configured audience model associations per membership
Configured tables per account	Each supported Region: 250	Yes	Maximum number of configured tables created per account
Configured tables per protected query	Each supported Region: 15	Yes	Maximum number of configured tables in a protected query
ID mapping tables per membership	Each supported Region: 5	Yes	Maximum number of ID mapping tables per membership
ID namespace associations per membership	Each supported Region: 10	Yes	Maximum number of ID namespace associations per membership

Name	Default	Adjusse	Description
Intermediate tables per membership	Each supported Region: 25	Yes	Maximum number of intermediate tables per membership
Members invited per collaboration	Each supported Region: 5	Yes	Maximum number of members invited per collaboration
Memberships per account	Each supported Region: 500	Yes	Maximum number of memberships per account
Privacy budget templates for access budgets per collaboration	Each supported Region: 25	Yes	Maximum number of privacy budget templates for access budgets per collaboration
Query text length (using differential privacy)	Each supported Region: 8 Kilobytes	No	Maximum text length for a SQL query statement using differential privacy
Query text length on Clean Rooms SQL analytics engine	Each supported Region: 90 Kilobytes	No	Maximum text length for a SQL query statement on Clean Rooms SQL analytics engine
Query text length on Spark analytics engine	Each supported Region: 500 Kilobytes	No	Maximum text length for a SQL query statement on Spark analytics engine
Rate of BatchGetCollaborationAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of BatchGetCollaborationAnalysisTemplate API calls per second

Name	Default	Adjusse	Description
Rate of BatchGetSchema requests	Each supported Region: 5 per second	Yes	Maximum number of BatchGetSchema API calls per second
Rate of BatchGetSchemaAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of BatchGetSchemaAnalysisRule API calls per second
Rate of CreateAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of CreateAnalysisTemplate API calls per second
Rate of CreateCollaboration requests	Each supported Region: 5 per second	Yes	Maximum number of CreateCollaboration API calls per second
Rate of CreateCollaborationChangeRequest requests	Each supported Region: 5 per second	Yes	Maximum number of CreateCollaborationChangeRequest API calls per second
Rate of CreateConfiguredAudienceModelAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of CreateConfiguredAudienceModelAssociation API calls per second
Rate of CreateConfiguredTable requests	Each supported Region: 5 per second	Yes	Maximum number of CreateConfiguredTable API calls per second
Rate of CreateConfiguredTableAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of CreateConfiguredTableAnalysisRule API calls per second

Name	Default	Adjusse	Description
Rate of CreateConfiguredTableAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of CreateConfiguredTableAssociation API calls per second
Rate of CreateConfiguredTableAssociationAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of CreateConfiguredTableAssociationAnalysisRule API calls per second
Rate of CreateIdMappingTable requests	Each supported Region: 5 per second	Yes	Maximum number of CreateIdMappingTable API calls per second
Rate of CreateIdNamespaceAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of CreateIdNamespaceAssociation API calls per second
Rate of CreateMembership requests	Each supported Region: 5 per second	Yes	Maximum number of CreateMembership API calls per second
Rate of CreatePrivacyBudgetTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of CreatePrivacyBudgetTemplate API calls per second
Rate of DeleteAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteAnalysisTemplate API calls per second

Name	Default	Adjuse	Description
Rate of DeleteCollaboration requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteCollaboration API calls per second
Rate of DeleteConfiguredAudienceModelAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteConfiguredAudienceModelAssociation API calls per second
Rate of DeleteConfiguredTable requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteConfiguredTable API calls per second
Rate of DeleteConfiguredTableAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteConfiguredTableAnalysisRule API calls per second
Rate of DeleteConfiguredTableAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteConfiguredTableAssociation API calls per second
Rate of DeleteConfiguredTableAssociationAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteConfiguredTableAssociationAnalysisRule API calls per second
Rate of DeleteIdMappingTable requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteIdMappingTable API calls per second

Name	Default	Adjuse	Description
Rate of DeleteIdNamespaceAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteIdNamespaceAssociation API calls per second
Rate of DeleteMember requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteMember API calls per second
Rate of DeleteMembership requests	Each supported Region: 5 per second	Yes	Maximum number of DeleteMembership API calls per second
Rate of DeletePrivacyBudgetTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of DeletePrivacyBudgetTemplate API calls per second
Rate of GetAnalysisLogExport requests	Each supported Region: 5 per second	Yes	Maximum number of GetAnalysisLogExport API calls per second
Rate of GetAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of GetAnalysisTemplate API calls per second
Rate of GetCollaboration requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaboration API calls per second
Rate of GetCollaborationAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaborationAnalysisTemplate API calls per second

Name	Default	Adjusse	Description
Rate of GetCollaborationChangeRequest requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaborationChangeRequest API calls per second
Rate of GetCollaborationConfiguredAudienceModelAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaborationConfiguredAudienceModelAssociation API calls per second
Rate of GetCollaborationIdNamespaceAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaborationIdNamespaceAssociation API calls per second
Rate of GetCollaborationPrivacyBudgetTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of GetCollaborationPrivacyBudgetTemplate API calls per second
Rate of GetConfiguredAudienceModelAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of GetConfiguredAudienceModelAssociation API calls per second
Rate of GetConfiguredTable requests	Each supported Region: 20 per second	Yes	Maximum number of GetConfiguredTable API calls per second
Rate of GetConfiguredTableAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of GetConfiguredTableAnalysisRule API calls per second

Name	Default	Adjusse	Description
Rate of GetConfiguredTableAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of GetConfiguredTable Association API calls per second
Rate of GetConfiguredTableAssociationAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of GetConfiguredTable AssociationAnalysisRule API calls per second
Rate of GetIdMappingTable requests	Each supported Region: 5 per second	Yes	Maximum number of GetIdMappingTable API calls per second
Rate of GetIdNamespaceAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of GetIdNamespaceAssociation API calls per second
Rate of GetMembership requests	Each supported Region: 5 per second	Yes	Maximum number of GetMembership API calls per second
Rate of GetPrivacyBudgetTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of GetPrivacyBudgetTemplate API calls per second
Rate of GetProtectedJob requests	Each supported Region: 5 per second	Yes	Maximum number of GetProtectedJob API calls per second
Rate of GetProtectedQuery requests	Each supported Region: 20 per second	Yes	Maximum number of GetProtectedQuery API calls per second

Name	Default	Adjusse	Description
Rate of GetSchema requests	Each supported Region: 5 per second	Yes	Maximum number of GetSchema API calls per second
Rate of GetSchemaAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of GetSchemaAnalysisRule API calls per second
Rate of ListAnalysisLogExports requests	Each supported Region: 5 per second	Yes	Maximum number of ListAnalysisLogExports API calls per second
Rate of ListAnalysisTemplates requests	Each supported Region: 5 per second	Yes	Maximum number of ListAnalysisTemplates API calls per second
Rate of ListCollaborationAnalysisTemplates requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationAnalysisTemplates API calls per second
Rate of ListCollaborationChangeRequests requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationChangeRequests API calls per second
Rate of ListCollaborationConfiguredAudienceModelAssociations requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationConfiguredAudienceModelAssociations API calls per second

Name	Default	Adjusse	Description
Rate of ListCollaborationIdNamespaceAssociations requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationIdNamespaceAssociations API calls per second
Rate of ListCollaborationPrivacyBudgetTemplates requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationPrivacyBudgetTemplates API calls per second
Rate of ListCollaborationPrivacyBudgets requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborationPrivacyBudgets API calls per second
Rate of ListCollaborations requests	Each supported Region: 5 per second	Yes	Maximum number of ListCollaborations API calls per second
Rate of ListConfiguredAudienceModelAssociations requests	Each supported Region: 5 per second	Yes	Maximum number of ListConfiguredAudienceModelAssociations API calls per second
Rate of ListConfiguredTableAssociations requests	Each supported Region: 5 per second	Yes	Maximum number of ListConfiguredTableAssociations API calls per second
Rate of ListConfiguredTables requests	Each supported Region: 5 per second	Yes	Maximum number of ListConfiguredTables API calls per second

Name	Default	Adjusse	Description
Rate of ListIdMappingTables requests	Each supported Region: 5 per second	Yes	Maximum number of ListIdMappingTables API calls per second
Rate of ListIdNamespaceAssociations requests	Each supported Region: 5 per second	Yes	Maximum number of ListIdNamespaceAssociations API calls per second
Rate of ListMembers requests	Each supported Region: 5 per second	Yes	Maximum number of ListMembers API calls per second
Rate of ListMemberships requests	Each supported Region: 5 per second	Yes	Maximum number of ListMemberships API calls per second
Rate of ListPrivacyBudgetTemplates requests	Each supported Region: 5 per second	Yes	Maximum number of ListPrivacyBudgetTemplates API calls per second
Rate of ListPrivacyBudgets requests	Each supported Region: 5 per second	Yes	Maximum number of ListPrivacyBudgets API calls per second
Rate of ListProtectedJobs requests	Each supported Region: 5 per second	Yes	Maximum number of ListProtectedJobs API calls per second
Rate of ListProtectedQueries requests	Each supported Region: 5 per second	Yes	Maximum number of ListProtectedQueries API calls per second

Name	Default	Adjus e	Description
Rate of ListSchemas requests	Each supported Region: 5	Yes	Maximum number of ListSchemas API calls per second
Rate of ListTagsForResource requests	Each supported Region: 5 per second	Yes	Maximum number of ListTagsForResource API calls per second
Rate of PopulateIdMappingTable requests	Each supported Region: 5 per second	Yes	Maximum number of PopulateIdMappingTable API calls per second
Rate of PreviewPrivacyImpact requests	Each supported Region: 5 per second	Yes	Maximum number of PreviewPrivacyImpact API calls per second
Rate of StartAnalysisLogExport requests	Each supported Region: 5 per second	Yes	Maximum number of StartAnalysisLogExport API calls per second
Rate of StartProtectedJob requests	Each supported Region: 5 per second	Yes	Maximum number of StartProtectedJob API calls per second
Rate of StartProtectedQuery requests	Each supported Region: 5 per second	Yes	Maximum number of StartProtectedQuery API calls per second
Rate of TagResource requests	Each supported Region: 5 per second	Yes	Maximum number of TagResource API calls per second
Rate of UntagResource requests	Each supported Region: 5 per second	Yes	Maximum number of UntagResource API calls per second

Name	Default	Adjusse	Description
Rate of UpdateAnalysisTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateAnalysisTemplate API calls per second
Rate of UpdateCollaboration requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateCollaboration API calls per second
Rate of UpdateCollaborationChangeRequest requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateCollaborationChangeRequest API calls per second
Rate of UpdateConfiguredAudienceModelAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateConfiguredAudienceModelAssociation API calls per second
Rate of UpdateConfiguredTable requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateConfiguredTable API calls per second
Rate of UpdateConfiguredTableAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateConfiguredTableAnalysisRule API calls per second
Rate of UpdateConfiguredTableAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateConfiguredTableAssociation API calls per second

Name	Default	Adjusse	Description
Rate of UpdateConfiguredTableAssociationAnalysisRule requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateConfiguredTableAssociationAnalysisRule API calls per second
Rate of UpdateIdMappingTable requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateIdMappingTable API calls per second
Rate of UpdateIdNamespaceAssociation requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateIdNamespaceAssociation API calls per second
Rate of UpdatePrivacyBudgetTemplate requests	Each supported Region: 5 per second	Yes	Maximum number of UpdatePrivacyBudgetTemplate API calls per second
Rate of UpdateProtectedJob requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateProtectedJob API calls per second
Rate of UpdateProtectedQuery requests	Each supported Region: 5 per second	Yes	Maximum number of UpdateProtectedQuery API calls per second
Table associations per membership	Each supported Region: 100	Yes	Maximum number of table associations per membership

AWS Clean Rooms resource parameter limits

Resource	Default	Description
Query text length	90 KB	Maximum text length for a SQL query statement
Query text length (using differential privacy)	90 KB	Maximum text length for a SQL query statement using differential privacy
Query run time	24 hours	Maximum duration a query is run before timeout

AWS Clean Rooms ML quotas

Your AWS account has the default quotas, formerly referred to as limits, for AWS Clean Rooms ML.

To view service quotas for AWS Clean Rooms ML, do one of the following:

- Follow the steps at Viewing service quotas in the *Service Quotas User Guide* and select AWS Clean Rooms ML as the service.
- Refer to the AWS Clean Rooms ML service quotas in the *Amazon Web Services General Reference*.

To maintain the performance of the service and to ensure appropriate usage of AWS Clean Rooms ML, the default quotas assigned to an account might be updated depending on Regional factors, payment history, fraudulent usage, and/or approval of a quota increase request.

Name	Default	Adjus e	Description
Active audience export jobs per audience generation job	Each supported Region: 25	No	The maximum number of active audience export jobs for an audience generation job

Name	Default	Adjus e	Description
Active configured model algorithm associations per membership	Each supported Region: 1,000	Yes	The maximum number of active configured model algorithm associations per membership
Active configured model algorithms per membership	Each supported Region: 1,000	Yes	The maximum number of active configured model algorithms per membership
Active custom model input channels per membership	Each supported Region: 100	Yes	The maximum number of active custom model input channels per membership
Maximum number of active training instances per account	Each supported Region: 10	Yes	The maximum number of active training instances that can be used per account for creating trained models.
Maximum number of active/pending/in progress trained model versions per trained model	Each supported Region: 100	Yes	The maximum number of active/pending/in progress trained model versions that can be created per trained model.
Maximum number of input columns to generate synthetic data	Each supported Region: 1,000	No	The maximum number of input columns to generate synthetic data
Maximum number of input rows to generate synthetic data	Each supported Region: 2,500,000	No	The maximum number of input rows to generate synthetic data

Name	Default	Adjusse	Description
Maximum number of ml.c4.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c4.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c4.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c4.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c4.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c4.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c4.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c4.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5.18xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5.18xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5.2xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.c5.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5.9xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5.9xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5n.18xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5n.18xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5n.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5n.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5n.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5n.4xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjusted	Description
Maximum number of ml.c5n.9xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5n.9xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c5n.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c5n.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.2xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.c6i.32xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.32xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c6i.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c6i.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.16xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.c7i.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.c7i.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.large training instances that can be used per account for creating trained models.

Name	Default	Adjusted	Description
Maximum number of ml.c7i.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.c7i.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g4dn.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g4dn.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g4dn.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g4dn.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.4xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.g4dn.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g4dn.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g4dn.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.2xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.g5.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g5.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g5.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.16xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.g6.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6.xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjusted	Description
Maximum number of ml.g6e.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.4xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.g6e.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.g6e.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.g6e.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m4.10xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m4.10xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m4.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m4.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m4.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m4.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m4.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m4.4xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.m4.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m4.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m5.12xlarge training instances per account	Each supported Region: 3	Yes	The maximum number of ml.m5.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m5.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m5.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m5.2xlarge training instances per account	Each supported Region: 3	Yes	The maximum number of ml.m5.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m5.4xlarge training instances per account	Each supported Region: 3	Yes	The maximum number of ml.m5.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m5.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m5.large training instances that can be used per account for creating trained models.

Name	Default	Adjusse	Description
Maximum number of ml.m5.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m5.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.32xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.32xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.m6i.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.large training instances that can be used per account for creating trained models.
Maximum number of ml.m6i.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m6i.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.16xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.m7i.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.m7i.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.large training instances that can be used per account for creating trained models.

Name	Default	Adjusable	Description
Maximum number of ml.m7i.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.m7i.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p2.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p2.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p2.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p2.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p2.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p2.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p4d.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p4d.24xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjusted	Description
Maximum number of ml.p4de.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p4de.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p5.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p5.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.p5en.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.p5en.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.16xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.r5.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.large training instances that can be used per account for creating trained models.
Maximum number of ml.r5.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5.xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.r5d.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.8xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.r5d.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.large training instances that can be used per account for creating trained models.
Maximum number of ml.r5d.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r5d.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.12xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.12xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.16xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.16xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.24xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.24xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.2xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.r7i.48xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.48xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.4xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.4xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.8xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.8xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.large training instances that can be used per account for creating trained models.
Maximum number of ml.r7i.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.r7i.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.t3.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.t3.2xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.t3.large training instances per account	Each supported Region: 0	Yes	The maximum number of ml.t3.large training instances that can be used per account for creating trained models.
Maximum number of ml.t3.medium training instances per account	Each supported Region: 0	Yes	The maximum number of ml.t3.medium training instances that can be used per account for creating trained models.
Maximum number of ml.t3.xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.t3.xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.trn1.2xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.trn1.2xlarge training instances that can be used per account for creating trained models.
Maximum number of ml.trn1.32xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.trn1.32xlarge training instances that can be used per account for creating trained models.

Name	Default	Adjus e	Description
Maximum number of ml.trn1n.32xlarge training instances per account	Each supported Region: 0	Yes	The maximum number of ml.trn1n.32xlarge training instances that can be used per account for creating trained models.
Maximum unique values in categorical prediction column	Each supported Region: 100	No	The maximum unique values in categorical prediction column
Minimum number of input columns required to generate synthetic data	Each supported Region: 5	No	The minimum number of input columns required to generate synthetic data
Minimum number of input rows required to generate synthetic data	Each supported Region: 1,500	No	The minimum number of input rows required to generate synthetic data
Pending/In-progress audience export jobs per customer	Each supported Region: 20	No	The maximum number of pending/in-progress audience export jobs per customer
Pending/In-progress audience generation jobs per customer	Each supported Region: 10	Yes	The maximum number of pending/in-progress audience generation jobs per customer
Pending/In-progress audience models per customer	Each supported Region: 2	Yes	The maximum number of pending/in-progress audience model training jobs per customer

Name	Default	Adjus e	Description
Pending/In-progress custom model inference jobs per account	Each supported Region: 10	Yes	The maximum number of pending/in-progress custom model inference jobs per account
Pending/In-progress custom model inference jobs per membership	Each supported Region: 5	Yes	The maximum number of pending/in-progress custom model inference jobs per membership
Pending/In-progress custom model training jobs per account	Each supported Region: 10	Yes	The maximum number of pending/in-progress custom model training jobs per account
Pending/In-progress custom model training jobs per membership	Each supported Region: 5	Yes	The maximum number of pending/in-progress custom model training jobs per membership
Pending/In-progress synthetic MLIC generation jobs per account	Each supported Region: 2	Yes	The maximum number of pending/in-progress synthetic MLIC generatio n jobs per account

AWS Clean Rooms ML has additional quotas listed below

Clean Rooms ML quotas

Resource	Default	Description
Maximum duration of a custom modeling inference job	25 hours	

Resource	Default	Description
Maximum number of interactions	20 billion	Maximum number of interactions allowed in training data. Larger inputs are sampled down.
Minimum number of interactions	1 million	
Maximum number of distinct users for lookalike model training	100 million	If more are included, only the top 100 million are used, ranked by number of interactions.
Minimum number of distinct users for lookalike model training	100,000	
Minimum number of users for export lookalike segment (audience) job	10,000	
Maximum number of distinct items used for model training.	1 million	You can include up to 50 million items, but only the most popular 1 million are used.
Maximum number of feature columns in the training dataset.	10	
Minimum number of distinct items per user	2	AWS Clean Rooms ML requires that each row or user has two or more items, including repeated items.
Maximum size of the seed audience	500,000	

Resource	Default	Description
Minimum size of the seed audience	500	The training data provider can set this value to as low as 25.
APIs	*per customer*	
Total number of active training datasets	500	
Total number of active lookalike models (audience models)	500	
Total number of active configured lookalike models (audience models)	10,000	
Total number of completed lookalike segment (audience) generation jobs	No limit	
Total number of completed export lookalike segment (audience) jobs	No limit	
Maximum duration of a lookalike model (audience model) generation job	25 hours	

Resource	Default	Description
Maximum duration of a lookalike segment (audience) generation job	10 hours	After you provide a seed, Clean Rooms ML takes a maximum of 10 hours to generate a lookalike segment. If you use an SQL query as the seed data, the query can take up to 24 hours to run before segment generation begins. A lookalike segment generation job times out after 35 hours, including the time taken by the seed query.
Maximum duration of a lookalike segment (audience) export job	1 hour	After you start an export, Clean Rooms ML takes a maximum of 1 hour to export a lookalike segment to your Amazon S3 bucket.
Minimum percentage for a segment (audience) size bin	1%	
Maximum percentage for a segment (audience) size bin	20%	
Minimum absolute size for a segment (audience) size bin	1% of the number of distinct users	
Maximum absolute size for a segment (audience) size bin	20% of the number of distinct users	

Clean Rooms ML API throttling quotas

Your AWS account has the following API transaction per second (TPS) per account per endpoint quotas.

Resource	Rate limit	Description
Rate of `CreateAudienceModel` requests	1 TPS rate, 3 TPS burst	Maximum number of `CreateAudienceModel` API calls per second
Rate of `CreateConfiguredAudienceModel` requests	10 TPS	Maximum number of `CreateConfiguredAudienceModel` API calls per second
Rate of `CreateTrainingDataset` requests	10 TPS	Maximum number of `CreateTrainingDataset` API calls per second
Rate of `DeleteAudienceGenerationJob` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteAudienceGenerationJob` API calls per second
Rate of `DeleteAudienceModel` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteAudienceModel` API calls per second
Rate of `DeleteConfiguredAudienceModel` requests	10 TPS	Maximum number of `DeleteConfiguredAudienceModel` API calls per second
Rate of `DeleteConfiguredAudienceModelPolicy` requests	25 TPS	Maximum number of `DeleteConfiguredAudienceModelPolicy` API calls per second
Rate of `DeleteTrainingDataset` requests	10 TPS	Maximum number of `DeleteTrainingDataset` API calls per second

Resource	Rate limit	Description
Rate of `GetAudienceGenerationJob` requests	50 TPS	Maximum number of `GetAudienceGenerationJob` API calls per second
Rate of `GetAudienceModel` requests	50 TPS	Maximum number of `GetAudienceModel` API calls per second
Rate of `GetConfiguredAudienceModel` requests	50 TPS	Maximum number of `GetConfiguredAudienceModel` API calls per second
Rate of `GetConfiguredAudienceModelPolicy` requests	50 TPS	Maximum number of `GetConfiguredAudienceModelPolicy` API calls per second
Rate of `GetTrainingDataset` requests	50 TPS	Maximum number of `GetTrainingDataset` API calls per second
Rate of `ListAudienceExportJobs` requests	50 TPS	Maximum number of `ListAudienceExportJobs` API calls per second
Rate of `ListAudienceGenerationJobs` requests	50 TPS	Maximum number of `ListAudienceGenerationJobs` API calls per second
Rate of `ListAudienceModels` requests	50 TPS	Maximum number of `ListAudienceModels` API calls per second

Resource	Rate limit	Description
Rate of `ListConfiguredAudienceModels` requests	50 TPS	Maximum number of `ListConfiguredAudienceModels` API calls per second
Rate of `ListTagsForResource` requests	50 TPS	Maximum number of `ListTagsForResource` API calls per second
Rate of `ListTrainingDatasets` requests	50 TPS	Maximum number of `ListTrainingDatasets` API calls per second
Rate of `PutConfiguredAudienceModelPolicy` requests	25 TPS	Maximum number of `PutConfiguredAudienceModelPolicy` API calls per second
Rate of `StartAudienceExportJob` requests	1 TPS rate, 3 TPS burst	Maximum number of `StartAudienceExportJob` API calls per second
Rate of `StartAudienceGenerationJob` requests	1 TPS rate, 5 TPS burst	Maximum number of `StartAudienceGenerationJob` API calls per second
Rate of `TagResource` requests	10 TPS	Maximum number of `TagResource` API calls per second
Rate of `UntagResource` requests	50 TPS	Maximum number of `UntagResource` API calls per second

Resource	Rate limit	Description
Rate of `UpdateConfiguredAudienceModel` requests	10 TPS	Maximum number of `UpdateConfiguredAudienceModel` API calls per second
Rate of `CreateConfiguredModelAlgorithm` requests	10 TPS	Maximum number of `CreateConfiguredModelAlgorithm` API calls per second.
Rate of `CreateConfiguredModelAlgorithmAssociation` requests	10 TPS	Maximum number of `CreateConfiguredModelAlgorithmAssociaton` API calls per second.
Rate of `PutMLConfiguration` requests	10 TPS	Maximum number of `PutMLConfiguration` API calls per second.
Rate of `CreateTrainedModel` requests	1 TPS rate, 3 TPS burst	Maximum number of `CreateTrainedModel` API calls per second.
Rate of `StartTrainedModelExportJob` requests	10 TPS	Maximum number of `StartTrainedModelExportJob` API calls per second.
Rate of `StartTrainedModelInferenceJob` requests	1 TPS rate, 3 TPS rate	Maximum number of `StartTrainedModelInferenceJob` API calls per second.

Resource	Rate limit	Description
Rate of `GetConfiguredModelAlgorithm` request	50 TPS	Maximum number of `GetConfiguredModelAlgorithm` API calls per second.
Rate of `GetConfiguredModelAlgorithmAssociation` request	50 TPS	Maximum number of `GetConfiguredModelAlgorithmAssociaton` API calls per second.
Rate of `GetTrainedModel` requests	50 TPS	Maximum number of `GetTrainedModel` API calls per second.
Rate of `GetMLConfiguration` requests	50 TPS	Maximum number of `GetMLConfiguration` API calls per second.
Rate of `GetTrainedModelInferenceJob` requests	50 TPS	Maximum number of `GetTrainedModelInferenceJob` API calls per second.
Rate of `ListConfiguredModelAlgorithm` requests	50 TPS	Maximum number of `ListConfiguredModelAlgorithm` API calls per second.
Rate of `ListConfiguredModelAlgorithmAssociations` requests	50 TPS	Maximum number of `ListConfiguredModelAlgorithmAssociatons` API calls per second.
Rate of `ListTrainedModels` requests	50 TPS	Maximum number of `ListTrainedModels` API calls per second.

Resource	Rate limit	Description
Rate of `ListCollaborationTrainedModelExportJobs` requests	50 TPS	Maximum number of `ListCollaborationTrainedModelExportJobs` API calls per second.
Rate of `ListCollaborationTrainedModelInferenceJobs` requests	50 TPS	Maximum number of `ListCollaborationTrainedModelInferenceJobs` API calls per second.
Rate of `DeleteConfiguredModelAlgorithm` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteConfiguredModelAlgorithm` API calls per second.
Rate of `DeleteConfiguredModelAlgorithmAssociation` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteConfiguredModelAlgorithmAssociaton` API requests per second.
Rate of `DeleteMLConfiguration` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteMLConfiguration` API requests per second.
Rate of `DeleteTrainedModelOutput` requests	2 TPS rate, 10 TPS burst	Maximum number of `DeleteTrainedModelOutput` API requests per second.

Document history for the AWS Clean Rooms User Guide

The following table describes the documentation releases for AWS Clean Rooms.

For notification about updates to this documentation, you can subscribe to the RSS feed. To subscribe to RSS updates, you must have an RSS plug-in enabled for the browser you are using.

Change	Description	Date
Added guidance for CSV headers in split inference requests	Added guidance about CSV header behavior in split inference requests. When an input CSV file is larger than the configured `maxPayloadInMB` value, SageMaker AI sends the CSV header row only in the first request. Your inference container must accept requests with or without a header row. For more information, see Model authoring guidelines for the inference container.	August 20, 2026
Support for minimum aggregation thresholds and comparison controls in custom analysis rules	You can now configure minimum aggregation thresholds and comparison controls in a custom analysis rule. A minimum aggregation threshold requires each result row to represent at least a minimum number of distinct data subjects. With comparison controls, you specify which columns a query can compare against a literal value and which columns it	August 13, 2026

	can compare against another column. For more informati on, see Minimum aggregation thresholds and Comparison controls.	
Export query analysis logs	You can now export redacted Spark logs for a completed SQL query to your own Amazon S3 bucket to troubleshoot failures and analyze query performan ce. To export logs, your membership needs the `CAN_EXPORT_QUERY_A NALYSIS_LOG` member ability. All collaboration members must approve the change request that grants it. For more information, see Exporting query analysis logs.	August 11, 2026
Corrected lookalike segment generation and export durations	Corrected the maximum time an SQL seed query can run for a lookalike segment (audience) generation job from 12 hours to 24 hours. Also added the 35-hour end-to-end maximum for a generatio n job and the maximum duration of an export job. For more information, see AWS Clean Rooms ML quotas.	August 11, 2026

Updated supported Spark properties for SQL queries and PySpark jobs	Added the Spark properties that can be configured when running SQL queries and PySpark jobs, and corrected the names of two propertie s. For more information, see Running SQL queries.	August 4, 2026
Clarified complex data type limitation for Amazon Athena data sources	Clarified that complex (nested) data types—ARRAY, MAP, and STRUCT—are not supported for Amazon Athena data sources. For more information, see Preparing data tables in Amazon Athena. Also corrected the procedure for preparing Amazon Athena data tables by removing a cryptographic computing step that doesn't apply to Amazon Athena data sources.	July 23, 2026
Added CR.8X worker compute type for SQL queries	Added the CR.8X worker compute type and updated worker specifications for SQL queries. For more informati on about running SQL queries in AWS Clean Rooms, see Running SQL queries.	July 11, 2026

User guide update for intermediate tables launch	Added documentation for intermediate tables, including creating, populating, configuring analysis rules, and deleting intermediate tables. For more information, see Intermediate tables in AWS Clean Rooms.	June 30, 2026
Added mutable payer configuration support	Added payer selection for compute costs when running queries, jobs, and ML workloads. Added new change request types for updating payer configurations and ML member abilities. Added new topic for updating a membership.	May 21, 2026
Updated Spark properties documentation	Updated Spark properties tables for SQL analysis templates and added Spark properties configuration to PySpark analysis templates.	April 17, 2026
Updated Documentation for Cost Allocation Tagging Section	Added documentation for AWS Billing and Cost Management and created section for cost allocation related taggable resources.	February 20, 2026

Support for cleanrooms-ml spark compute configuration	Customers can now customize Spark runtime behavior and optimize SQL performance by configuring supported Spark properties for SQL queries. This capability is now available for audience generation job creation and MLInputChannel creation.	January 5, 2026
Support for detailed monitoring with CloudWatch	Customers can now opt in to detailed monitoring with CloudWatch metrics for operational monitoring of collaborations, including query performance and resource utilization.	January 2, 2026
Support for Collaboration Change Request EventBridge notification events	Added new EventBridge events for Collaboration Change Request lifecycle events.	December 19, 2025
Support for change requests to manage collaboration membership and settings	Collaboration creators can now submit change requests to add new members, update existing member abilities, and modify auto-approval settings. All collaboration members must approve change requests before changes take effect.	December 18, 2025

End of support timeline for AWS Clean Rooms SQL analytics engine	Support for the AWS Clean Rooms SQL analytics engine is ended on December 17, 2025. Customers can now use Spark SQL analytics engine only.	December 17, 2025
Support for additional EventBridge notification events	Added new EventBridge events for Analysis Template, Configured Table Associati on, Configured Table Associati on Analysis Rule, Id Mapping Table, Id Namespace Associati on, Protected Job, and Protected Query.	December 15, 2025
Support for privacy-enhanced synthetic dataset generation for training custom machine learning models	Customers can now generate privacy-enhanced synthetic datasets to train custom machine learning models while protecting sensitive data.	November 30, 2025
Support for configurable Spark properties in SQL queries	Customers can now customize Spark runtime behavior by configuring supported Spark properties for SQL queries.	October 30, 2025
Support for cross-Region collaboration	AWS Clean Rooms now supports cross-Region collaboration for AWS Glue and Athena data tables, allowing customers to process data stored in one AWS Region and deliver results to Regions approved by all collaboration members.	October 3, 2025

Support for data access budgets	Customers can now set and manage data access budgets to control table usage.	October 1, 2025
Populate ID mapping table – update	Customers can now use incremental processing in a rule-based ID mapping workflow to more efficient ly process large datasets. Customers can also delete records from an ID mapping workflow to help comply with data management regulatio ns.	September 23, 2025
Support for variable workload sizes for PySpark jobs	Customers can now choose and adjust their compute size to run PySpark jobs more effectively.	September 4, 2025
Support for adding new members to an existing collaboration	Customers can now add new members to existing collaborations. This feature includes options to configure automatic approval settings and block membership requests.	September 3, 2025
Added support for configuri ng error summaries and redaction for custom model logs	Customers can now configure error summaries and redaction for custom model logs in configured model algorithm association.	September 3, 2025

Support for Snowflake authentication option	Customers can now use key-pair authentication with a Snowflake PEM private key when storing their Snowflake credentials in AWS Secrets Manager for AWS Clean Rooms connections.	August 29, 2025
Added error configuration for PySpark analysis templates	Customers can now configure basic or detailed error message levels in PySpark analysis templates to manage error visibility and troublesh ooting capabilities.	August 19, 2025
Support for EventBridge	Customers can now use Amazon EventBridge to route events from AWS Clean Rooms to other software components.	July 31, 2025
Update to existing policy	The following new permissio ns have been added to the `AWSCleanRoomsFullA ccessNoQuerying` managed policy: `cleanroom s:UpdateConfigured TableAllowedColumns` and `cleanrooms:UpdateC onfiguredTableRefe rence` .	July 29, 2025
Increased editing capabilities for configured tables	Customers can now modify allowed columns and table reference fields for configured tables.	July 29, 2025

ML input channel supports Parquet file type	Customers can now choose the Parquet file type as the data format the ML input channel can use.	July 17, 2025
Support for differential privacy on Spark SQL analytics engine	Customers can now use differential privacy with the Spark SQL analytics engine. In addition, the maximum text length for a SQL query statement using differential privacy is now 90 KB.	July 16, 2025
End of support timeline for AWS Clean Rooms SQL analytics engine	Support for new collaborations using the Clean Rooms SQL analytics engine ends on July 16, 2025. As of July 17, 2025, you can't create new collaborations with the legacy Clean Rooms SQL analytics engine. AWS Clean Rooms ends support for the Clean Rooms SQL analytics engine on December 17, 2025.	July 16, 2025
Support for incremental and distributed training	Customers can now use incremental training and distributed training for their custom models.	July 1, 2025

End of support timeline for Clean Rooms SQL analytics engine	Starting now through July 16, 2025, new collaborations using the legacy Clean Rooms SQL analytics engine require a limit increase. Starting July 17, 2025, you can't create new collaborations with this engine. AWS Clean Rooms ends support for the Clean Rooms SQL analytics engine on December 17, 2025.	May 22, 2025
Update to PySpark analysis templates	Customers can now name a user script file with any valid Python filename.	May 15, 2025
Support for multiple result receivers	Customers can now select multiple members to receive query results from a single query.	April 30, 2025
Support for migrating collaborations to Spark SQL	AWS Clean Rooms SQL now supports aggregation and list analysis rules, in addition to custom analysis rules. Additionally, customers can update an existing collabora tion to use the Spark analytics engine which powers Spark SQL.	April 2, 2025
Support for PySpark jobs	Customers can now analyze data by running jobs using approved PySpark analysis templates.	March 18, 2025

Update to existing policies	The following new permission has been added to the `AWSCleanRoomsMLReadOnlyAccess` managed policy: `PassCleanRoomsResources`. The following new permissions have been added to the `AWSCleanRoomsMLFullAccess` managed policy: `PassCleanRoomsResources` and `ConsoleDescribeECRRepositories`.	January 10, 2025
Support for multiple compute workers	Customers can now specify what type of compute workers and how many to provision when creating a lookalike segment.	December 17, 2024
Support for multiple data sources and clouds	Customers can now use multiple data sources and clouds, such as Amazon Athena and Snowflake, to collaborate with their partners' datasets.	December 1, 2024
Clean Rooms ML Custom Modeling is now available	Customers can now use their own custom ML models in a collaboration.	November 7, 2024
New analytics engine	Customers who have large datasets can now run complex queries using SQL functions supported by the Spark SQL analytics engine.	October 29, 2024

Enhanced privacy protection, build lookalike audiences, choose multiple result receivers	You can protect your data while also allowing complex activation queries using **Additional analyses** and the **collaboration analysis rule**. You can build lookalike audience models from SQL queries or analysis templates. You can choose multiple members to receive results.	July 24, 2024
Entity resolution in AWS Clean Rooms	With AWS Entity Resolution in AWS Clean Rooms, you can create an ID mapping table between two ID namespaces to query events data across disparate identity spaces.	July 23, 2024
Update to existing policy	The following new permission has been added to the `AWSCleanRoomsFullAccessNoQuerying` managed policy: `cleanrooms:BatchGetSchemaAnalysisRule` .	May 13, 2024
AWS Clean Rooms ML is now fully available	AWS Clean Rooms ML provides a privacy-enhancing method for two parties to identify similar users in their data without the need to share their data with each other.	April 3, 2024

Update to existing policy	The Statement ID in the `AWSCleanRoomsFullAccess` managed policy has been updated from ConsolePickQueryResultsBucket to SetQueryResultsBucket to better represent the permissions since the permissions.	March 21, 2024
New managed policies for AWS Clean Rooms ML	Two new managed policies have been added: `AWSCleanRoomsMLReadOnlyAccess` and `AWSCleanRoomsMLFullAccess`.	November 29, 2023
AWS Clean Rooms ML (preview)	AWS Clean Rooms ML provides a privacy-enhancing method for two parties to identify similar users in their data without the need to share their data with each other.	November 29, 2023
AWS Clean Rooms Differential Privacy (preview)	Customers can now use AWS Clean Rooms Differential Privacy to help protect the privacy of their users.	November 29, 2023
Payment configuration	The collaboration creator can now configure either the member who can run queries or a different member in the collaboration to be billed for query compute costs.	November 14, 2023

Query run time - update	The maximum duration a query is run before timeout has been updated from 4 hours to 12 hours.	October 6, 2023
CloudFormation resources - update	AWS Clean Rooms has added the following new resources : `AWS::CleanRooms::Membership ProtectedQueryOutputConfiguration` , `AWS::CleanRooms::Membership ProtectedQueryResultConfiguration` , and `AWS::CleanRooms::Membership ProtectedQueryS3OutputConfiguration` .	September 7, 2023
CloudFormation resources - update	AWS Clean Rooms has added the following new resources : `AWS::CleanRooms::AnalysisTemplate` and `AWS::CleanRooms::ConfiguredTableAnalysisRuleCustom` .	August 31, 2023

Separate member abilities	The collaboration creator can now designate one member as the member who can query and another member as the member who can receive results. This gives the collaboration creator the ability to make sure that the member who can query doesn't have access to the query results.	August 30, 2023
AWS Clean Rooms Glossary	Documentation-only update to add a glossary of AWS Clean Rooms terms.	August 30, 2023
Support for Apache Iceberg tables (preview)	AWS Clean Rooms now supports Apache Iceberg tables (preview).	August 25, 2023
Quotas update	The Quotas section has been updated to reflect the new default quota for membershi ps per account.	August 9, 2023

Update to existing policy	The following new permissions have been added to the `AWSCleanRoomsFullAccessNoQuerying` managed policy: `cleanrooms:CreateAnalysisTemplate`, `cleanrooms:GetAnalysisTemplate`, `cleanrooms:UpdateAnalysisTemplate`, `cleanrooms:DeleteAnalysisTemplate`, `cleanrooms:ListAnalysisTemplates`, `cleanrooms:GetCollaborationAnalysisTemplate`, `cleanrooms:BatchGetCollaborationAnalysisTemplate`, and `cleanrooms:ListCollaborationAnalysisTemplates`.	July 31, 2023
Analysis templates and Custom analysis rule	AWS Clean Rooms now supports analysis templates and the **Custom** analysis rule. Analysis templates enable collaborators to build or import their own custom SQL query to use in the collaboration. With the **Custom** analysis rule, the table owner can approve custom SQL queries on their configured tables.	July 31, 2023

Analysis rules support the OR logical condition	AWS Clean Rooms analysis rules now support the OR logical condition in the JOIN clause.	June 29, 2023
CloudFormation integration	AWS Clean Rooms now integrates with CloudFormation.	June 15, 2023
Analysis builder	Members who can query and receive results now have the ability to run queries on some tables without writing SQL code by using the **Analysis builder UI**.	June 15, 2023
SQL functions	Documentation-only update to clarify supported SQL functions.	May 5, 2023
Troubleshooting	Documentation-only update to add a Troubleshooting section for common issues.	April 27, 2023
Supported data types for AWS Clean Rooms	Documentation-only update to add a new section that lists supported AWS Glue Data Catalog data types.	April 26, 2023
Examples of AWS CloudTrail events	Documentation-only update to add examples of CloudTrail events for StartProtectedQuery (successful) and StartProtectedQuery (failed).	April 20, 2023

Update to existing policy	The following new permissions have been added to the `AWSCleanRoomsFullAccessNoQuerying` managed policy: `cleanrooms:ListTagsForResource` , `cleanrooms:UntagResource` , and `cleanrooms:TagResource` . For more information, see AWS managed policies.	March 21, 2023
General availability	AWS Clean Rooms is now generally available.	March 21, 2023
Preview release	Preview release of the AWS Clean Rooms User Guide	January 12, 2023

AWS Clean Rooms Glossary

Consult this glossary to become familiar with terminology that is used for AWS Clean Rooms.

Aggregation analysis rule

The query restriction that allows queries that aggregate analysis using COUNT, SUM, or AVG functions along optional dimensions. These queries won't reveal row-level information.

Supports use cases such as campaign planning, media reach, frequency, and conversion measurement.

Other types of analysis rules are custom and list.

Analysis rules

The query restrictions that authorize a specific type of query.

The analysis rule type determines what kind of analysis can be run on the configured table. Each type has a predefined query structure. You control how your table columns can be used in the structure through the query controls.

The types of analysis rules are aggregation, list, and custom.

Analysis template

A collaboration-specific, pre-approved query that can be reused.

Supported formats: SQL code or Python code for Spark.

If using SQL, the analysis template can contain parameters wherever a literal value could typically appear in a SQL query. For more information about supported parameter types, see Data types in the *AWS Clean Rooms SQL Reference*.

Analysis templates only work with the custom analysis rule.

Base table

Any configured table, ID mapping table, or other intermediate table that is used in the analysis to populate an intermediate table.

C3R encryption client

The Cryptographic Computing for Clean Rooms (C3R) encryption client.

Used to encrypt and decrypt data, C3R is a client-side encryption SDK with a command line interface.

Cleartext column

A column that isn't cryptographically protected for either a JOIN or SELECT SQL construct.

Cleartext columns can be used in any part of the SQL query.

Collaboration

A secure logical boundary in AWS Clean Rooms in which members can perform SQL queries on configured tables.

Collaborations are created by the collaboration creator.

Only members who have been invited to the collaboration can join the collaboration.

A collaboration can have only one member who can query data or one member who can run queries and jobs.

A collaboration can have only one member who can receive results.

A collaboration can have one or more members paying for query compute costs or members paying for query and job compute costs.

All members can see the list of invited participants in the collaboration before they join the collaboration.

Collaboration creator

The member who creates a collaboration.

There is only one collaboration creator per collaboration.

Only the collaboration creator can remove members from the collaboration or delete the collaboration.

Configured table

Each configured table represents a reference to an existing table in the AWS Glue Data Catalog that has been configured for use in AWS Clean Rooms. A configured table contains an analysis rule that determines how the data can be used.

Currently, AWS Clean Rooms supports associating data stored in Amazon Simple Storage Service (Amazon S3) that is cataloged through AWS Glue.

For more information about AWS Glue, see the AWS Glue Developer Guide.

Configured tables can be associated to one or more collaborations.

> **Note**
>
> AWS Clean Rooms doesn't currently support Amazon S3 bucket locations that are registered with AWS Lake Formation.

Custom analysis rule

The query restriction that allows a specific set of pre-approved queries (analysis templates) or allows a specific set of accounts that can provide queries or jobs that use your data.

Supports use cases such as first-touch attribution, incremental analyses, and audience discovery analyses.

Supports differential privacy.

Other types of analysis rules are aggregation and list.

Decryption

The process of transforming encrypted data back to its original form. Decryption can only be performed if you have access to the secret key.

Differential privacy

A mathematically-rigorous technique that protects the collaboration data from the member who can receive results learning about a specific individual.

Encryption

The process of encoding data into a form that appears random using a secret value called a key. It's impossible to determine the original plaintext without access to the key.

Fingerprint column

A column that is cryptographically protected for a JOIN SQL construct.

ID mapping workflow method

How you want the ID mapping to be performed.

There are two ID mapping workflow methods:

- Rule-based ID mapping – The method by which you use matching rules to translate first-party data from a source to a target in an ID mapping workflow.
- Provider services ID mapping – The method by which you use a provider service to translate third party-encoded data from a source to a target in an ID mapping workflow.

  AWS Clean Rooms currently supports LiveRamp as the provider services-based ID mapping workflow method. You must have a subscription to LiveRamp through AWS Data Exchange to use this method. For more information, see Subscribe to a provider service on AWS Data Exchange in the *AWS Entity Resolution User Guide.*

ID mapping table

A resource in AWS Clean Rooms that enables either first-party matching rules or multi-party identity transcoding in a collaboration.

A ID mapping table is a reference to an existing table in the AWS Glue Data Catalog. It contains an ID mapping table analysis rule that determines how the data can be queried in AWS Clean Rooms. ID mapping tables can be associated to one or more collaborations.

ID mapping table analysis rule

A type of analysis rule managed by AWS Clean Rooms and used to join disparate identity data to facilitate querying. It's automatically added to ID mapping tables and can't be edited. It inherits

the behaviors of the other analysis rules in the collaboration – as long as those analysis rules are homogeneous.

ID mapping workflow

A data processing job that maps data from a source to a target based on the specified ID mapping workflow method. It produces an ID mapping table.

ID namespace

A resource in AWS Clean Rooms that contains metadata explaining datasets across multiple AWS accounts and how to use these datasets in an ID mapping workflow.

ID namespace association

An association of an ID namespace resource that helps you discover inputs into their ID mapping workflow.

Intermediate table

A collaboration-scoped resource in AWS Clean Rooms that caches analysis results within a collaboration for reuse in subsequent analyses. Intermediate tables store data in AWS managed storage and are not exported outside the collaboration.

Job

A method to access and analyze configured tables in a collaboration using a supported set of functions, classes, and variables.

AWS Clean Rooms currently supports the PySpark job type.

AWS Clean Rooms currently supports running jobs using a PySpark analysis template.

List analysis rule

The query restriction that allows queries that output row-level attribute analysis of the overlap between this table and the tables of the member who can query.

Supports use cases such as enrichment and audience building or suppression.

Other types of analysis rules are aggregation and custom.

Lookalike model

A model of a training data provider's data that allows a seed data provider to create a lookalike segment of training data provider's data that most closely resembles their seed data.

Lookalike segment

A subset of the training data that most closely resembles the seed data.

Member

An AWS customer who is a participant in a collaboration.

A member is identified using their AWS account.

All members can contribute data.

Member who can query

The member who can query data in the collaboration.

There is only one member who can query per collaboration, and that member is immutable.

An administrative user can use AWS Identity and Access Management (IAM) permissions to control which of their IAM principals (such as users or roles) can query data in the collaboration. For more information, see Create a service role to read data from Amazon S3.

Member who can run queries and jobs

The member who can run queries and jobs on the data in the collaboration.

There is only one member who can run queries and jobs per collaboration, and that member is immutable.

An administrative user can use AWS Identity and Access Management (IAM) permissions to control which of their IAM principals (such as users or roles) can run queries and jobs in the collaboration. For more information, see Create a service role to read data from Amazon S3.

Member who can receive results

A member who can receive query results. A member who can receive results specifies query results settings for the Amazon S3 destination and the query result format (CSV or Parquet).

For analyses with the Spark analytics engine, a member who can receive results also specifies whether files should be output as a single file or as multiple files.

There can be more than one member who can receive results in a collaboration.

Member paying for query compute costs

A member who is responsible for paying for query compute costs.

One or more members can be designated as payer candidates for query compute costs in a collaboration. When running a query, the member who initiates the query selects which payer candidate pays for the compute costs.

The collaboration creator can update payer configurations after the collaboration is created by submitting a change request.

The member paying for query compute costs receives a bill for the queries that have been run in the collaboration.

Member paying for query and job compute costs

A member who is responsible for paying for query and job compute costs.

One or more members can be designated as payer candidates for query and job compute costs in a collaboration. When running a query or job, the member who initiates selects which payer candidate pays for the compute costs.

The collaboration creator can update payer configurations after the collaboration is created by submitting a change request.

The member paying for query and job compute costs receives a bill for the queries that have been run in the collaboration.

Membership

A resource created when a member joins a collaboration.

All resources that the member associates to a collaboration are a part of the membership or are associated with the membership.

Only the member that owns the membership can add, remove, or edit resources in that membership.

Sealed column

A column that is cryptographically protected for a SELECT SQL construct.

Seed data

The seed data provider's data, which is used to create a lookalike segment. The seed data can be provided directly or it can come from the results of an AWS Clean Rooms query. The lookalike segment output is a set of users from the training data that most closely resembles the seed users.

Spark analytics engine

An analytics option in AWS Clean Rooms that enables customers to run complex queries on large datasets stored in Amazon S3, Amazon Athena, or Snowflake using Apache Spark SQL functions. It also supports PySpark analysis in AWS Clean Rooms.

When you create a collaboration using the CreateCollaboration API, the value of the Spark analytics engine is SPARK.

Query

A method to access and analyze configured tables in a collaboration, using a supported set of functions, classes, and variables.

AWS Clean Rooms currently supports the SQL query language.

AWS Clean Rooms currently supports running direct SQL queries or running queries using a SQL analysis template.